Optimisation intégrée et interactive pour l'agencement d'espace

Julien Bénabès

École Centrale de Nantes

ÉCOLE DOCTORALE
SCIENCES POUR L'INGÉNIEUR, GÉOSCIENCES, ARCHITECTURE

Année 2011 N° B.U. :

Thèse de DOCTORAT

Spécialité : GÉNIE MÉCANIQUE

Présentée et soutenue publiquement par :

JULIEN BÉNABÈS

le 05 décembre 2011
à l'École Centrale de Nantes

OPTIMISATION INTÉGRÉE ET INTERACTIVE POUR L'AGENCEMENT D'ESPACE

JURY

Président :	Bernard YANNOU	*Professeur, LGI, École Centrale de Paris, Châtenay-Malabry, France*
Rapporteurs :	Georges FADEL	*Professeur, Clemson University, Clemson, États-Unis*
	Jean-Pierre NADEAU	*Professeur, TREFLE, École Nationale Supérieure d'Arts et Métiers, Talence, France*
Examinateurs :	Yannick RAVAUT	*Ingénieur-Docteur, Thales Communications & Security, Cholet, France*
	Fouad BENNIS	*Professeur, IRCCyN, École Centrale de Nantes, Nantes, France*
	Émilie POIRSON	*Maître de Conférences, IRCCyN, École Centrale de Nantes, Nantes, France*

Directeur de thèse : Fouad BENNIS
Co-encadrante : Émilie POIRSON
Laboratoire : Institut de Recherche en Communications et Cybernétique de Nantes, IRCCyN, UMR CNRS 6597

N° ED. : 498-202

Remerciements

Ce travail de doctorat a été réalisé au sein de l'Institut de Recherche en Communications et Cybernétique de Nantes (IRCCyN) en partenariat avec l'École Centrale de Nantes. Je tiens donc en premier lieu à remercier Michel Malabre, directeur de l'IRCCyN et Patrick Chedmail, directeur de l'École Centrale de Nantes pour m'avoir accueilli dans leur établissement respectif et permis de réaliser mon projet de recherche dans les meilleures conditions.

Je tiens très sincèrement à remercier également mes encadrants de thèse Fouad Bennis, professeur à l'École Centrale de Nantes, et Émilie Poirson, maître de conférences à l'École Centrale de Nantes. Je suis très reconnaissant de leur disponibilité, leur confiance et leur implication tout au long de ces trois années de doctorat.

J'adresse également mes remerciements aux membres de mon jury de thèse : Georges Fadel et Jean-Pierre Nadeau qui ont accepté d'être rapporteurs de ma thèse, Bernard Yannou qui a présidé ce jury de thèse et Yannick Ravaut qui m'a fait l'honneur de compléter ce jury.

Ces travaux de doctorat n'auraient pas été aussi concrets sans des cas d'application industriels. Ces cas d'étude, proposés par Yannick Ravaut et Ulrika Dabadie, ingénieurs de l'entreprise Thales Communications & Security à Cholet, m'ont grandement aidé lorsqu'il a fallu matérialiser et valider les différents concepts développés dans cette thèse.

Un grand merci aux chercheurs, ingénieurs et techniciens que j'ai pu côtoyer pendant ce doctorat, que ce soit dans mes activités de recherche au sein des équipes MCM et IS3P de l'IRCCyN, dans mes activités d'enseignement à l'École Centrale de Nantes ou encore dans mes activités associatives. Vos remarques, critiques et conseils m'ont été fort utiles tant pour l'aboutissement de ce projet que pour ma propre construction professionnelle.

Enfin, je remercie chaleureusement tous mes collègues doctorants et amis du laboratoire que j'ai côtoyés au cours de ces trois années : Nicolas, Semaan, Aude, Guillaume, Benoît, Roland, Jad, Coralie, David, Raphaël et Yu. J'ai partagé avec certains un bureau de l'IRCCyN et avec d'autres des pauses-café très conviviales. Je remercie chaleureusement mon amie, mon fils, ma famille et mes amis qui m'ont encouragé dans ce projet.

Il est certain que j'oublie certaines personnes... qu'ils m'en excusent. Ces trois années m'ont permis de rencontrer beaucoup de personnes qui ont de près ou de loin contribué à ce travail. Ils se reconnaîtront.

Le progrès en sciences provient toujours d'une combinaison
de pensées décousues et de pensées rigoureuses...
cette combinaison est notre outil le plus précieux.

Gregory Bateson (1904–1980).
Anthropologue, psychologue et épistémologue américain.

Table des matières

Introduction

L'optimisation d'agencement d'espace est actuellement une thématique de recherche qui intéresse de nombreux scientifiques et qui trouve de nombreuses applications dans le milieu industriel, notamment pour les problèmes de chargement de containers, le placement de machines dans une usine ou encore l'agencement de pièces dans un assemblage mécanique.

Cette problématique fait donc l'objet de nombreuses études car l'agencement d'espace est une tâche complexe. En effet, dans la plupart des applications industrielles, les exigences du concepteur sont nombreuses et la recherche de solutions optimales au problème d'optimisation nécessite très souvent beaucoup de temps de calcul et de ressources informatiques. L'objectif principal des stratégies de résolution – actuellement en cours de développement – est donc de proposer au concepteur une grande diversité de solutions optimales, dans des temps de calcul raisonnables. Ces méthodes sont le plus souvent dédiées à un cas d'application spécifique et peuvent difficilement être adaptées à de nombreux problèmes. D'autres recherches se penchent donc aujourd'hui sur le caractère générique de ces méthodes d'optimisation d'agencement.

Dans de nombreuses applications, l'optimisation d'agencement est considérée comme une activité multidisciplinaire qui faire intervenir un grand nombre de concepteurs provenant de métiers divers. Ces concepteurs ont leurs propres exigences de conception et leur propre expertise sur le problème d'agencement. Une problématique majeure de la résolution de ces problèmes réside donc dans la formulation du problème d'optimisation d'agencement. Cette formulation doit traduire au mieux toutes les exigences du ou des concepteurs. Lorsque certaines de ces exigences ne peuvent pas être formulées simplement, il convient alors de faire participer le concepteur à la recherche de solutions optimales, tout au long du processus d'optimisation. Cette interactivité avec le concepteur permet notamment d'intégrer le jugement personnel de ce dernier dans la construction d'un agencement « idéal ».

Ce doctorat s'inscrit dans la continuité des travaux initiés par Guillaume Jacquenot, doctorant CIFRE (Jac10) dans le cadre du projet MDO (Multidisciplinary Design Optimization), labellisé par le pôle de compétitivité EMC2 (Ensembles Métalliques et Composites Complexes). L'objectif de ce projet MDO, qui s'est déroulé de 2006 à 2010, était de diffuser les techniques d'optimisation et de conception multidisciplinaire dans le milieu industriel. Dans ce projet, une action de recherche particulière sur la problématique d'optimisation d'agencement d'espace a été menée conjointement entre l'ECN (École Centrale de Nantes), l'IRCCyN (Institut de Recherche en Communications et

Cybernétique de Nantes) et l'entreprise SIREHNA.

Ce manuscrit de thèse décrit donc les travaux de recherche réalisés dans ce doctorat. Ces travaux proposent une méthode d'optimisation d'agencement d'espace, qui possède les trois caractéristiques suivantes :

- méthode **intégrée** : l'approche proposée vise à fournir au concepteur un ensemble de méthodes et d'outils qui lui permettent de décrire, formuler et résoudre le problème et prendre une décision sur les solutions optimales proposées par l'algorithme d'optimisation,
- méthode **générique** : la méthode doit pouvoir s'adapter au plus grand nombre de problèmes d'agencement,
- méthode **interactive** : l'approche a pour objectif de pouvoir intégrer le concepteur dans le processus d'optimisation, afin que celui-ci puisse apporter son expertise et son jugement personnel à la génération de solutions optimales.

Ces travaux de recherche s'intéressent donc dans un premier temps à la manière de décrire et formuler de manière générique un problème d'optimisation d'agencement. L'objectif est ici de construire un modèle géométrique du problème et d'associer à ce modèle toutes les exigences du concepteur afin de transformer le problème d'agencement en un problème d'optimisation d'agencement. Ensuite, ces travaux ont contribué à la mise en œuvre d'une stratégie d'optimisation modulaire, basée sur la combinaison entre un algorithme génétique et des modules d'optimisation locale. Ces travaux se sont également orientés vers le développement d'approches interactives qui permettent au concepteur de faire des choix de conception parmi les nombreuses solutions proposées par l'algorithme d'optimisation. Ces méthodes consistent donc à trier les solutions, les visualiser et les comparer dans des espaces multidimensionnels et les explorer en prenant en compte les perceptions du concepteur. L'objectif de cette interactivité est d'améliorer encore les solutions proposées par l'algorithme en ajoutant à ces solutions le jugement personnel du concepteur.

Ce manuscrit de thèse est donc structuré en plusieurs chapitres de la manière suivante. Le chapitre 1 réalise un état de l'art des recherches effectuées en optimisation, notamment sur les problématiques d'agencement d'espace. Ce chapitre décrit également les objectifs de recherche fixés dans le doctorat. Le chapitre 2 est dédié aux méthodes qui permettent de décrire et formuler de manière générique un problème d'optimisation d'agencement. Le chapitre 3 présente la stratégie d'optimisation modulaire qui permet de résoudre ces problèmes. Le chapitre 4 propose un ensemble de méthodes et d'outils qui permettent au concepteur de faire des choix par rapport aux solutions générées par l'algorithme d'optimisation. Le chapitre 5 présente des applications industrielles de la démarche d'optimisation proposée dans ce manuscrit. Au travers des cas d'étude présentés, la méthode, dans sa globalité, est appliquée et les résultats sont discutés. Dans ce même chapitre, une partie est dédiée à la présentation du démonstrateur qui a été développé au cours de ce doctorat et qui matérialise la méthode d'optimisation d'agencement qui est proposée dans ce manuscrit.

1 État de l'art et objectifs de recherche

1.1 Introduction

Ce premier chapitre présente l'état de l'art relatif aux travaux de recherche effectués dans ce doctorat. Cette bibliographie propose, dans un premier temps, quelques fondamentaux sur l'optimisation de conception. Une attention particulière est notamment apportée à l'optimisation multiobjectif. La formulation des problèmes et les stratégies de résolution sont discutées.

Dans un second temps, nous nous intéresserons plus particulièrement à l'agencement d'espace. Après une brève définition générale de la problématique, nous détaillerons les différentes approches qui permettent de décrire et formuler ces problèmes d'optimisation. L'objectif de cette partie est éga-

lement de proposer un aperçu général des algorithmes de résolution utilisés dans la recherche de solutions optimales. Cette partie décrit notamment une stratégie d'optimisation hybride, appliquée au placement d'objets à l'intérieur d'un coffre de voiture.

Ensuite, la quatrième partie de ce chapitre présente quelques méthodes permettant d'intégrer le concepteur dans le processus d'optimisation. En effet, dans les problèmes complexes d'ingénierie, l'interactivité avec le concepteur est une étape essentielle du processus de résolution du problème. Un des objectifs de cette interactivité est notamment d'associer l'expertise, le jugement personnel et les perceptions du concepteur à la construction d'une solution « idéale ». La notion de solution idéale caractérise ici une solution qui répond à la fois aux critères d'optimisation quantitatifs et aux préférences qualitatives exprimées par le concepteur. Cette partie est notamment illustrée par des applications d'agencement d'espace qui font intervenir le concepteur dans le processus d'optimisation.

Enfin, la dernière partie de ce chapitre précise les objectifs de recherche fixés dans ce doctorat et présente d'un point de vue systémique, la méthode d'optimisation proposée afin de résoudre de manière générique les problèmes d'agencement d'espace.

1.2 Optimisation de conception

Cette première partie a pour objectif de rappeler les fondamentaux de l'optimisation de conception. Nous nous intéresserons plus particulièrement aux concepts et méthodes liés à l'optimisation multiobjectif.

1.2.1 Définition de l'optimisation de conception

L'optimisation de conception peut être assimilée à l'ensemble des méthodes et outils permettant au concepteur d'améliorer un produit ou un système. Ces méthodes visent à rechercher les meilleures solutions réalisant le meilleur compromis entre tous les critères de performance du produit étudié. Une solution est une combinaison de variables et de paramètres de conception qui définissent le produit. Les variables de conception sont également appelés variables d'optimisation et définissent l'ensemble des paramètres sur lesquels le concepteur peut agir pour améliorer son produit. Une solution est également caractérisée par des contraintes de conception et des **c**ritères de performances relatifs au cahier des charges du produit. Dans la suite de ce manuscrit, le terme anglais *design* pourra être utilisé pour représenter une solution.

L'optimisation de conception est actuellement un domaine de recherche qui fait l'objet de nombreuses études. En effet, les industriels sont de plus en plus nombreux à mettre en œuvre, dans leur entreprise, une démarche d'optimisation car ils cherchent continuellement à améliorer le triplet « coût, qualité, délai ». Ce besoin industriel s'explique par diverses raisons, dont notamment :
– la forte compétitivité mondiale entre les entreprises,

– l'évolution rapide des technologies et des systèmes de production,
– l'amélioration des interactions avec le client.

Ainsi, pour le concepteur, les avantages liés à l'utilisation d'une méthode d'optimisation sont diverses :
– rechercher des alternatives (solutions différentes) innovantes répondant au cahier des charges du produit,
– trouver des solutions qui réalisent le meilleur compromis en termes de performances et d'exigences de conception,
– justifier, auprès du décideur, ses choix technologiques par des données quantitatives sur les performances et les contraintes liées au problème.

En résumé, une méthode d'optimisation consiste premièrement à décrire et formuler le problème (créer un modèle) en traduisant au mieux toutes les exigences du concepteur. Ensuite, la démarche suggère au concepteur l'utilisation d'une stratégie d'optimisation appropriée. Enfin, la dernière étape consiste à prendre une décision, en termes de choix de conception, par rapport aux différentes alternatives optimales proposées par l'algorithme.

1.2.2 Formulation d'un problème d'optimisation

D'une manière générale, tout problème d'optimisation, quel qu'il soit, est toujours formulé mathématiquement de la manière suivante :

$$\text{Problème P} : \begin{cases} \text{Trouver les valeurs optimales des variables de conception } \mathbf{x}^* \\ \text{avec } \mathbf{x} = (x_1, x_2, ..., x_n) \in \mathbb{R}^n \\ \mathbf{x}^* \in \operatorname{argmin} \mathbf{F}(\mathbf{x}, \mathbf{p}) = \operatorname{argmin} \left(f_1(\mathbf{x}, \mathbf{p}), f_2(\mathbf{x}, \mathbf{p}), ...f_m(\mathbf{x}, \mathbf{p})\right) \\ \text{sous contraintes :} \\ \mathbf{h}(\mathbf{x}^*, \mathbf{p}) = 0 \\ \mathbf{g}(\mathbf{x}^*, \mathbf{p}) \leqslant 0 \end{cases} \tag{1.1}$$

où n représente le nombre de variables d'optimisation et m le nombre d'objectifs. Le vecteur $\mathbf{x}$ définit les variables de conception qui sont modifiables. La valeur de ces variables est fixée à la fin du processus d'optimisation. Le vecteur $\mathbf{p}$, généralement omis dans la formulation du problème, représente l'ensemble des paramètres qui ne sont pas modifiés au cours du processus d'optimisation. Le vecteur de fonctions $\mathbf{F}$ regroupe les critères d'optimisation. Les vecteurs de fonctions $\mathbf{h}$ et $\mathbf{g}$ définissent respectivement les contraintes d'égalité et d'inégalité du problème. Ces deux fonctions définissent l'ensemble des variables qui répondent au cahier des charges du problème, c'est-à-dire l'espace de conception du problème d'optimisation. En résumé, si une solution appartient à l'espace de conception du problème, cette solution est une solution admissible. Nous utiliserons également le terme de solution faisable.

Dans la formulation définie par l'équation 1.1, lorsque $m = 1$, on dit que le problème d'optimisation est mono-objectif alors que dans les autres cas, le problème est multiobjectif. Les problèmes mono-objectif sont, dans la plupart des cas, plus simples à résoudre. Le problème consiste à trouver une solution optimale qui respecte toutes les contraintes de conception. Dans les problèmes d'optimisation multiobjectif, les critères de performances sont multiples et bien souvent contradictoires. L'objectif est alors de trouver une ou plusieurs solutions qui minimisent ces critères. Il existe différentes manières d'exprimer le minimum d'un ensemble de critères d'optimisation (Ehr05). Dans la partie suivante, dédiée à ces problèmes multiobjectif, une définition de ce minimum est présentée : la Pareto-dominance.

1.2.3 Optimisation multiobjectif

1.2.3.1 Définition

Un problème d'optimisation multiobjectif a pour but de minimiser simultanément plusieurs critères définis dans le même espace. Par exemple, de nombreuses conceptions de produits ou systèmes cherchent à minimiser le coût de leur production tout en maximisant leur performance. Ces deux critères d'optimisation sont contradictoires et la solution du problème est un compromis entre les deux objectifs, comme l'illustre la figure 1.1. La formulation multiobjectif d'un problème d'optimisation est donnée par l'équation 1.1, avec $m > 1$.

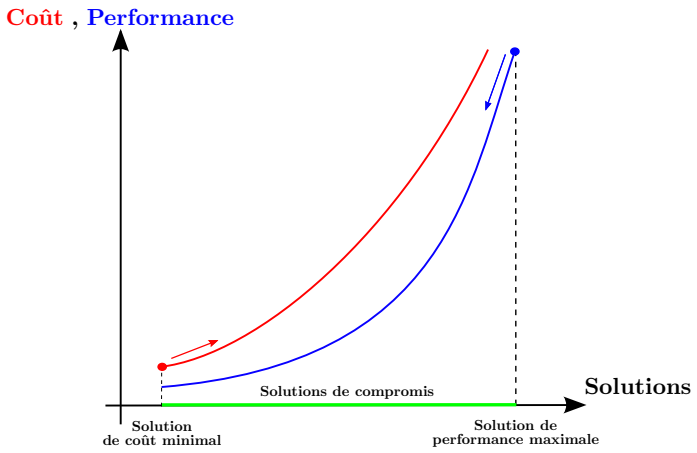


FIGURE 1.1 – *Illustration graphique d'un problème d'optimisation multiobjectif*

Sur la figure 1.1, on remarque que lorsque la performance du produit augmente, le coût de conception augmente également et inversement. Le concepteur doit donc trouver une solution qui réalise un compromis entre ces deux critères de performances du produit.

1.2.3.2 Pareto dominance

Dans un problème d'optimisation multiobjectif, le concepteur doit comparer des solutions qui sont caractérisées par plusieurs performances. Le travail du concepteur revient donc à comparer deux vecteurs de valeurs numériques au lieu de deux simples valeurs dans le cas mono-objectif. Pour comparer ces deux vecteurs, la notion de dominance au sens de Pareto (Par96) est utilisée. On admet alors que $\mathbf{F}(x)$ domine $\mathbf{F}(y)$ au sens de Pareto (ou $\mathbf{F}(x) \preceq \mathbf{F}(y)$), si et seulement si :

$$\begin{cases} F_i(x) \leqslant F_i(y) \, \forall i \in \{1, ..., m\} \\ \text{et} \\ \exists \, i_0 \in \{1, ..., m\} \, , \, F_{i_0}(x) < F_{i_0}(y) \end{cases} \tag{1.2}$$

On dira de la même manière qu'une solution solution x domine une solution y, si $\mathbf{F}(x)$ domine $\mathbf{F}(y)$ au sens de Pareto. En utilisant cette notion de Pareto dominance, il est possible de définir la notion de front de Pareto comme l'ensemble des points Pareto-optimaux (optimaux au sens de Pareto), c'est à dire l'ensemble des solutions non-dominées par une autre solution de l'espace de conception. Si un *design* est Pareto-optimal, il est impossible de trouver un autre *design* qui soit au moins aussi bon sur tous les objectifs et strictement meilleur sur au moins un de ceux-ci. La figure 1.2 illustre les quatre fronts de Pareto possibles pour un problème d'optimisation à deux objectifs. L'espace d'arrivée du problème est représenté en gris. Les lignes en trait épais noir représentent les différents fronts de Pareto.

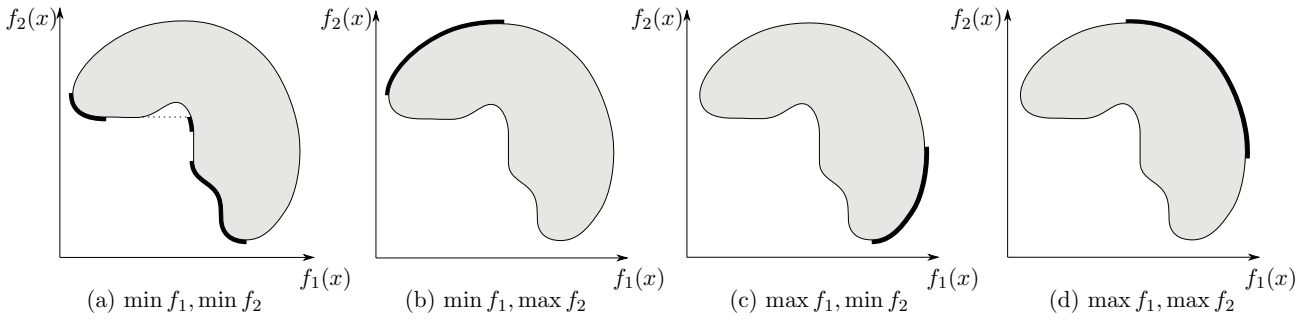


FIGURE 1.2 – *Fronts de Pareto de problèmes d'optimisation à deux objectifs. Image empruntée à (Jac10)*

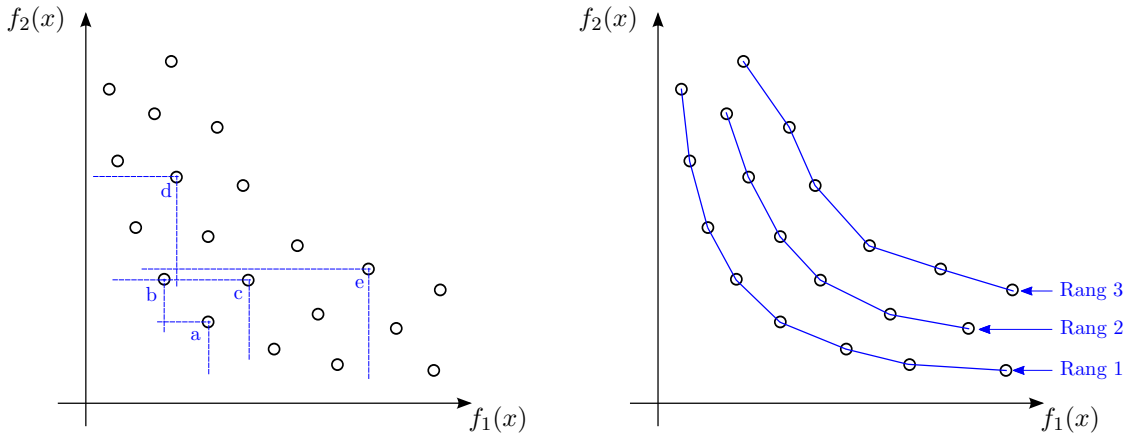
Pour un problème d'optimisation multiobjectif, il existe un unique front de Pareto de solutions. Cependant, l'algorithme d'optimisation utilisé pour résoudre ce problème ne trouve pas systématiquement l'intégralité de ce front de Pareto. Dans la plupart des cas, l'algorithme génère un ensemble de solutions proches de ce front de Pareto. Cependant, dans le suite de ce manuscrit, nous considèrerons que ces solutions optimales obtenues par l'algorithme forment le front de Pareto du problème d'optimisation. Ce front de Pareto sera également appelé frontière de Pareto ou surface de compromis. Aussi, une solution Pareto-optimale d'un problème d'optimisation multiobjectif sera assimilée

à une solution optimale dans la suite du document.

Pour classer les *designs* d'un problème multiobjectif, on utilise la notion de rang. Le rang traduit la position d'un *design* dans un ensemble de solutions. Il existe plusieurs définitions de rang, les plus connus étant le rang de Fonseca et Fleming et le rang de Goldberg :

– rang de Fonseca et Fleming (FF93) : le rang d'un *design* est défini comme le nombre de *designs* le dominant plus 1,
– rang de Goldberg (Gol89a) : le rang d'un *design* est défini comme le nombre de fronts le dominant plus 1.

Les solutions du front de Pareto ont un rang égal à un. Le reste des solutions ont alors un rang supérieur ou égal à 2. La figure 1.3 (a) illustre cette notion de rang de Fonseca et Fleming pour un problème de minimisation à deux objectifs ($\min f_1$ et $\min f_2$). Chaque cercle représente un *design* dans l'espace des objectifs. Les *designs* a et b ont un rang égal à 1, les *designs* c et d ont un rang égal à 3, et le *design* e a un rang égal à 7. La figure 1.3 (b) illustre la notion de rang de Goldberg avec le même problème à deux objectifs.



(a) rang de Fonseca et Fleming

(b) rang de Goldberg

FIGURE 1.3 – *Illustration de la notion de rang de Fonseca/Fleming et de Goldberg*

Cette notion de rang est utilisée dans le chapitre 4 relatif à la prise de décision faite par le concepteur à la fin du processus d'optimisation. Le rang est alors considéré comme un indicateur de comparaison de solutions générées par l'algorithme d'optimisation.

1.2.3.3 Stratégies de résolution

Classification des méthodes

Il existe un nombre important de méthodes de résolution des problèmes d'optimisation multiobjectif. Ces méthodes peuvent être classées de différentes façons, selon la manière dont le concepteur va prendre en compte les critères d'optimisation dans la résolution du problème. Par exemple, Van

Veldhuizen *et al.* (VV99) propose de trier les méthodes en 3 grandes familles de stratégies :
- les méthodes de résolution **à priori** : le concepteur définit, avant d'exécuter l'algorithme, ses préférences sur les différents objectifs d'optimisation,
- les méthodes de résolution **progressive** : le concepteur définit ses préférences sur les objectifs au fur et à mesure de l'avancement du calcul,
- les méthodes de résolution **à posteriori** : le concepteur choisit une solution de compromis parmi les meilleures alternatives proposées par l'algorithme.

Aussi, Colette *et al.* propose de classer ces mêmes méthodes d'optimisation en cinq catégories (CS02) :
- **les méthodes scalaires** : ces méthodes consistent à transformer le problème d'optimisation multiobjectif en un problème d'optimisation mono-objectif. Ces méthodes sont des méthodes de résolution à priori. Trois de ces méthodes sont exposées dans le paragraphe suivant,
- **les méthodes interactives** : ces méthodes font partie de la famille des méthodes de résolution progressive. Le concepteur participe à la recherche d'une seule solution du problème d'optimisation. Plusieurs de ces méthodes interactives sont abordées dans la section 4 de ce chapitre, relative à l'optimisation interactive,
- **les méthodes floues** : ces méthodes d'optimisation utilisent le principe de la logique floue (Ver92). Parmi ces méthodes floues, nous pouvons citer la méthode de Sakawa ou encore la méthode de Reardon.
- **les méthodes exploitant une métaheuristique** : ces méthodes de résolution sont dédiées aux problèmes d'optimisation « difficiles » (SY99). Les métaheuristiques sont généralement des algorithmes stochastiques itératifs, qui progressent vers un optimum global. Les méthodes utilisant une métaheuristique les plus connues sont l'algorithme du recuit simulé (KGV83), la recherche tabou (GL97) et l'algorithme génétique (Deb98; Gol89a). L'algorithme génétique, utilisé dans la stratégie d'optimisation proposée dans ce manuscrit est expliqué dans la suite de ce chapitre,
- **les méthodes d'aide à la décision** : ces méthodes de résolution permettent d'obtenir un ensemble de solutions en fixant une relation d'ordre entre les différents objectifs du problème d'optimisation. Parmi ces méthodes d'aide à la décision, nous pouvons citer la méthode AHP (Saa77; Saa00), ELECTRE (Roy85) et PROMETHEE (BP85).

Cette classification, proposée par Colette *et al.* ne met pas en évidence que certaines stratégies d'optimisation peuvent se retrouver dans plusieurs catégories d'algorithmes. Dans le chapitre 4 de ce manuscrit, un algorithme génétique interactif, visant à explorer perceptivement un ensemble de solutions d'agencement, est décrit. Cet algorithme est basé à la fois sur l'utilisation d'une métaheuristique et sur l'interaction entre l'algorithme et le concepteur, afin que ce dernier exprime ses préférences sur les solutions qui lui sont présentées.

Méthodes d'optimisation scalaires

Ce paragraphe décrit brièvement trois méthodes d'optimisation scalaires, dédiées à la résolution des problèmes d'optimisation multiobjectif. Ces méthodes, qui proposent une approche de résolution à priori, consistent à ramener le problème multiobjectif à un problème mono-objectif. Ces trois méthodes sont la méthode de pondération (Coe00), la méthode des ε-contraintes (Mie99) (ou méthode du compromis) et la méthode du but à atteindre (Gem74).

La méthode de pondération consiste à réaliser une somme pondérée des différents objectifs. Le problème devient alors :

$$\begin{cases} \min\limits_{\mathbf{x}\in\mathbb{R}^n} F(\mathbf{x}) = \sum_{i=1}^{m} w_i.f_i(\mathbf{x}) \\ \sum_{i=1}^{m} w_i = 1 \,\text{et}\, w_i \geqslant 0 \end{cases} \tag{1.3}$$

Les poids w_i sont arbitrairement choisis par le concepteur. En donnant une valeur plus importante au poids w_i, la fonction f_i aura alors une influence plus importante dans la somme pondérée. Le plus souvent, il convient de résoudre plusieurs problèmes d'optimisation mono-objectif en considérant plusieurs combinaisons des w_i mais cette solution se révèle alors plus coûteuse en temps de calcul. La figure 1.4 illustre cette méthode pour un problème à deux objectifs (min f_1 et min f_2).

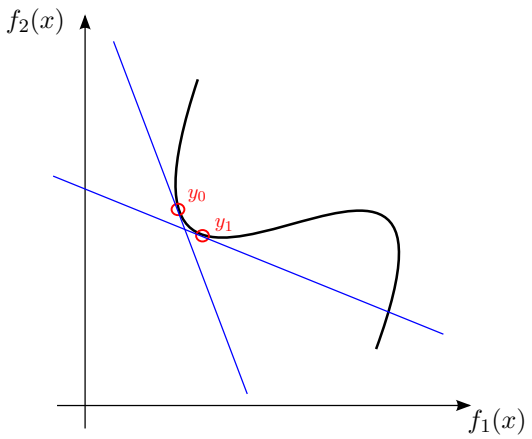


FIGURE 1.4 – *Méthode de pondération pour un problème à deux objectifs : min f_1 et min f_2*

Pour le problème à deux objectifs, illustré sur la figure 1.4, l'équation 1.3 devient : $f_2(\mathbf{x}) = \frac{1}{w_2}F(\mathbf{x}) - \frac{w_1}{w_2}f_1(\mathbf{x})$. Puisque l'on cherche à minimiser F, on cherche donc la droite de coefficient directeur $-\frac{w_1}{w_2}$ ayant la plus petite ordonnée à l'origine et qui soit tangente à l'ensemble des solutions Pareto-optimales.

L'utilisation de la méthode de pondération pose un problème car cette stratégie d'optimisation permet de détecter seulement les solutions qui se trouvent sur l'enveloppe convexe du front de Pareto (Geo68). L'utilisation de la méthode des ε-contraintes pallie ce problème. Dans la méthode des ε-contraintes, une des fonctions est considérée comme l'objectif du problème d'optimisation. Les

autres fonctions sont considérées comme des contraintes. Le problème devient donc :

$$\begin{cases} \min\limits_{\mathbf{x}\in\mathbb{R}^n} f_{i_0}(\mathbf{x}) \\ f_i(\mathbf{x}) \leqslant \varepsilon_i \text{ pour } i \neq i_0 \end{cases} \tag{1.4}$$

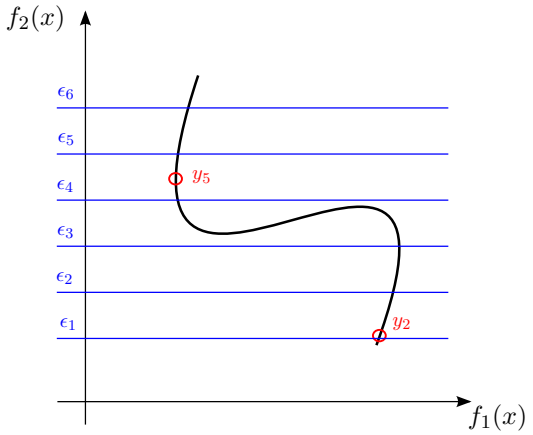


FIGURE 1.5 – *Méthode des ε-contraintes pour un problème à deux objectifs : min f_1 et min f_2*

Comme dans la méthode de pondération, il est possible de résoudre successivement plusieurs problèmes d'optimisation mono-objectif sous contraintes avec à chaque fois des ε_i différents. La figure 1.5 illustre cette méthode pour le même problème d'optimisation que précédemment.

Dans la méthode du but à atteindre, nous nous ramenons au problème d'optimisation mono-objectif suivant :

$$\begin{cases} \min\limits_{\mathbf{x}\in\mathbb{R}^n} \alpha \\ f_i(\mathbf{x}) - \alpha.d_i \leqslant z_i \text{ pour } i = \{1, ..., m\} \end{cases} \tag{1.5}$$

Dans cette formulation, z peut être vu comme un point de $\mathbb{R}^m$ et d comme un vecteur de $\mathbb{R}^m$, où m caractérise le nombre de critères d'optimisation. Avec cette méthode, un choix à priori reste à faire à la fois sur le point z et sur la direction d.

Pour un même point z, il s'agit alors de résoudre différents problèmes d'optimisation mono-objectif avec successivement différentes directions d. Il faut ensuite renouveler cette opération avec différents choix de z. Ce qui augmente bien plus encore les temps de calcul.

Une métaheuristique : l'algorithme génétique

L'algorithme génétique est un algorithme d'optimisation évolutionnaire qui a été développé à l'origine par John Holland dans les années 1970 à l'Université de Michigan aux États-Unis (Hol92). L'algorithme génétique cherche une solution approchée à un problème d'optimisation en imitant le processus naturel de l'évolution (Gol89b). L'algorithme génétique est une métaheuristique qui a pour objectif, au cours de son processus de calcul, d'améliorer une population courante de solutions.

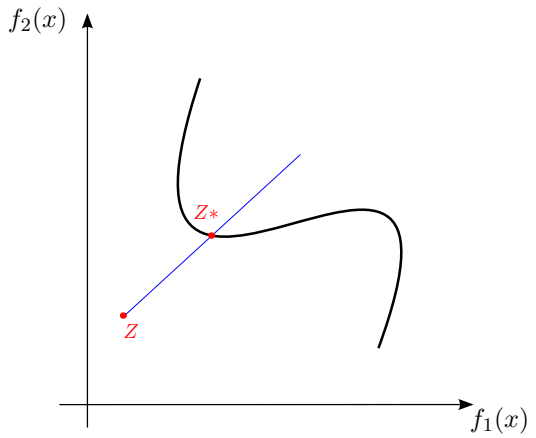


FIGURE 1.6 – *Méthode du but à atteindre pour un problème à deux objectifs : min f_1 et min f_2*

Ces solutions sont considérées comme les individus de la population. Les algorithmes génétiques sont des algorithmes d'optimisation très robustes car ils sont particulièrement adaptés aux problèmes où l'initialisation n'est pas intuitive, où les variables ne sont pas du même type (continu, discret,...) et où le critère d'optimisation n'est pas nécessairement évalué par une fonction mathématique explicite mais par un processus de calcul extérieur.

Le principe de fonctionnement général d'un algorithme génétique est illustré sur la figure 1.7. Considérons une population initiale d'individus, aléatoirement créés, choisis arbitrairement par le concepteur ou générés par un autre processus de calcul. Cette population est ensuite évaluée suivant les contraintes et les objectifs formulés dans le problème d'optimisation. Si les critères d'arrêt de l'algorithme sont satisfaits, l'algorithme s'arrête, sinon des opérateurs génétiques sont appliqués sur cette population afin de faire évoluer celle-ci faire une nouvelle population d'individus meilleurs, satisfaisant les exigences du problèmes. Les individus de la $(n+1)^{ème}$génération sont les « enfants » des individus de la $(n)^{ème}$génération, qui sont eux les « parents ».

Les opérateurs génétiques qui sont généralement utilisés dans l'algorithme génétique ont deux objectifs : faire converger la population courante d'individus vers un ensemble de solutions optimales (processus d'intensification) et explorer la plus grande proportion de l'espace de conception (processus de diversification). Ces opérateurs sont la sélection, la reproduction, le croisement et la mutation. La sélection consiste à choisir, parmi les individus d'une population donnée, certaines solutions qui paraissent meilleures selon les contraintes et objectifs définis par le concepteur. La méthode de sélection la plus connue est la méthode de sélection par roulette, où les individus qui ont de meilleures performances ont une probabilité plus importante d'être sélectionnés. Cette méthode de sélection a été améliorée pour donner la méthode de sélection universelle stochastique. L'opérateur de croisement consiste à croiser plusieurs individus entre eux afin de générer des individus meilleurs. L'opérateur de mutation consiste à modifier certaines variables d'un individu donné, afin de créer un nouvel individu différent et assurer ainsi que la recherche de solutions parcourt la plus grande partie possible de l'espace de conception. Chaque opérateur génétique a ses propres réglages

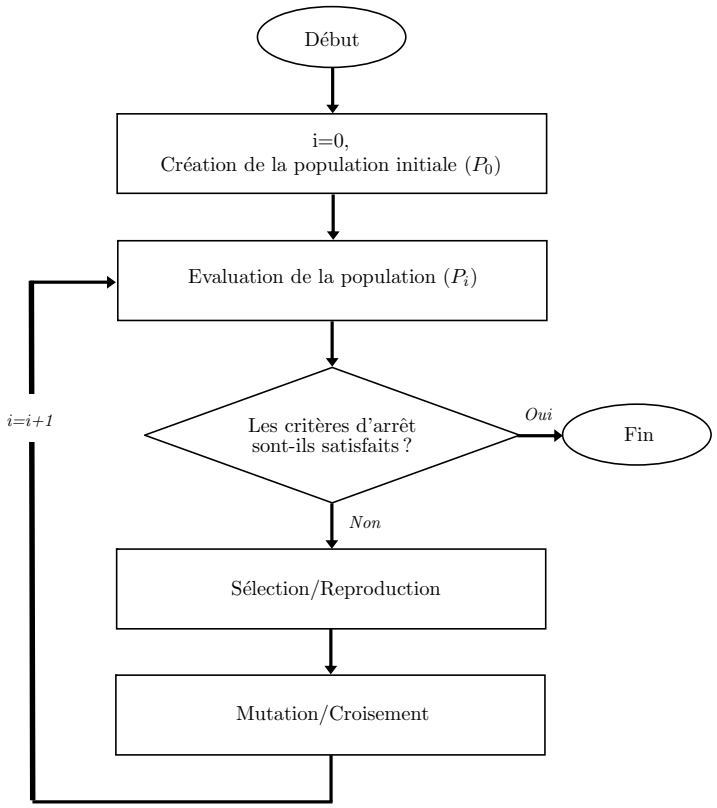


FIGURE 1.7 – *Fonctionnement général d'un algorithme génétique*

et l'efficacité de l'algorithme génétique dépend fortement de ces réglages. Plusieurs études sont notamment consacrées à l'influence de ces paramètres sur les performances de l'algorithme génétique (ZCZ05; HWLL02; JR08).

D'autres mécanismes peuvent aussi être intégrés à l'algorithme génétique. Par exemple, l'opérateur d'élitisme est utilisé afin de ne pas perdre les meilleurs individus générés au cours du processus d'optimisation. Une étude comparative de plusieurs algorithmes génétiques met en évidence l'influence de cet opérateur génétique sur les performances de l'algorithme (ZDT00). D'autres opérateurs de diversité garantissent le fait que l'algorithme génétique ne converge pas seulement vers une seule solution mais essaie de proposer un ensemble de solutions équitablement réparties sur la frontière de Pareto.

Les algorithmes génétiques multiobjectif les plus connus et utilisés sont le NSGA-II (DAPM00; DPAM02) et le SPEA2 (ZLT02). La plupart des nouvelles métaheuristiques à population sont comparées à ces deux algorithmes. La différence entre tous les algorithmes génétiques développés réside en particulier dans la manière dont sont implémentés les opérateurs génétiques. Finalement, l'algorithme génétique est une méthode d'optimisation très générique qui possède quelques avantages :

– une bonne adaptation aux problèmes d'optimisation difficiles,
– une solution au problème est toujours générée : cette solution s'améliore au fur et à mesure du

processus de résolution de l'algorithme génétique,
- la possibilité de stopper volontairement ou non le processus de recherche (panne machine ou interaction avec le concepteur par exemple) et de le reprendre ensuite,
- de nombreuses possibilités d'amélioration de ces algorithmes basées sur l'hybridation de l'algorithme avec d'autres procédures de calcul.

Actuellement, certaines recherches se portent sur l'amélioration des performances des algorithmes génétiques sur des problèmes de conception industriels. Certaines études proposent des stratégies d'optimisation hybride qui associent à l'algorithme génétique d'autres algorithmes d'optimisation. La stratégie d'optimisation modulaire, proposé dans le chapitre 3 du manuscrit, est basé sur ce principe. Enfin, d'autres recherches portent sur le développement de micro-algorithmes génétiques qui utilisent une population d'individus de petite taille permettant ainsi un faible nombre d'évaluations des fonctions objectif. AMGA (Archive-based Micro Genetic Algorithm) est un des exemples de micro-algorithmes génétiques développés récemment (TFD11).

1.2.3.4 Mesure des performances des algorithmes d'optimisation multiobjectif

Les résultats issus des différents algorithmes d'optimisation, présentés précédemment, sont les solutions Pareto optimales du problème d'optimisation étudié. Ces solutions forment la surface de compromis du problème multiobjectif et cette surface est idéalement confondue avec la frontière de Pareto du problème. Afin de comparer les performances des algorithmes d'optimisation, il convient donc de comparer la qualité des surfaces de compromis produits par ces mêmes algorithmes.

Pour comparer ces surfaces de compromis, un certain nombre d'indicateurs ont été définis. Ces indicateurs visent à fournir une mesure numérique caractéristique de la surface de compromis. Parmi ces caractéristiques, nous pouvons citer la proximité par rapport au front de Pareto, la qualité d'approximation du front de Pareto ou encore la diversité des solutions. En résumé, deux types d'indicateurs sont utilisés : les indicateurs unaires (ou absolus) et les indicateurs binaires (ou relatifs). Les indicateurs unaires analysent une surface de compromis en la comparant à la frontière de Pareto du problème d'optimisation. L'utilisation de tels indicateurs est possible si le front de Pareto ou une approximation de celui-ci est connu. Les indicateurs unaires les plus généralement utilisés sont :
- la mesure de la distance entre la surface de compromis et le front de Pareto : cette mesure est définie par une distance euclidienne, dans l'espace des objectifs, entre une solution de la surface de compromis et la solution la plus proche appartenant au front de Pareto du problème d'optimisation. Cette distance est moyennée sur l'ensemble des solutions de la surface de compromis,
- la mesure de la représentation au front de Pareto : cet indicateur évalue la portion du front de Pareto découvert par la surface de compromis. Le calcul de cet indicateur est très similaire au calcul de l'indicateur précédent,
- la métrique d'espacement S (Sch95) : cette métrique caractérise l'uniformité de la répartition des solutions composant la surface de compromis. La métrique d'espacement S mesure donc

l'écart-type entre les différentes distances des solutions de la surface de compromis au front de Pareto,

- la mesure de la diversité phénotypique (DAPM00) : cette mesure est basé sur la métrique d'espacement S mais intègre en plus les distances avec les solutions extrêmes du front de Pareto. Cette mesure permet de qualifier ainsi la quantité du front de Pareto approximé,
- la mesure de la distance phénotypique maximale (Zit99) : cette mesure évalue la longueur de la diagonale de l'hyperboîte définie par les extrêmes générés par l'algorithme d'optimisation,
- la mesure de l'hypervolume (ZDT00) : cet indicateur évalue le volume couvert par les solutions de la surface de compromis dans l'espace des objectifs. L'hypervolume est calculé comme l'union des différents hypercubes, construits pour chaque solution de la surface de compromis à partir d'un point de référence W de même dimension que l'espace des objectifs. La figure 1.8 illustre le calcul de l'hypervolume pour une série de points non dominés dans un problème de minimisation bi-objectif. Le cercle noir représente le point Nadir et le carré noir le point de référence W. Ce point de référence W est généralement choisi comme le point dominé par l'ensemble des points de la surface de compromis. Plus grand est l'hypervolume, meilleure est la qualité des points non dominés obtenus. L'évolution de l'hypervolume donne une bonne information sur l'évolution de la convergence de l'algorithme d'optimisation. Enfin, si le front de Pareto est connu, il est possible de normer l'hypervolume d'une surface de compromis par l'hypervolume du front de Pareto (VV99).

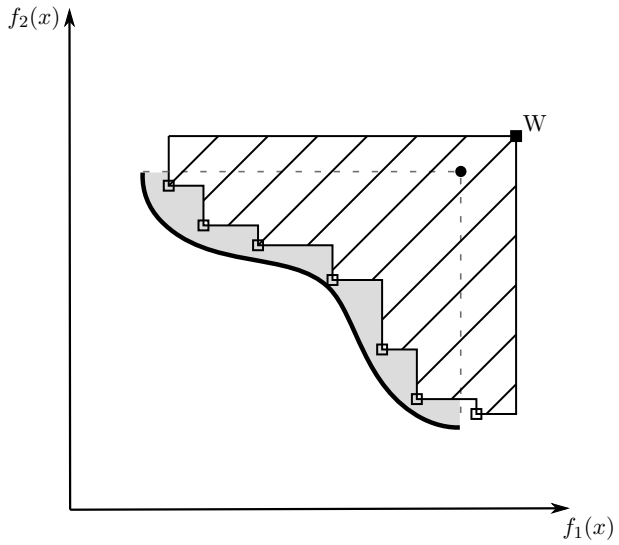


FIGURE 1.8 – *Calcul de l'hypervolume pour un problème de minimisation bi-objectif. Image empruntée à (Jac10)*

Les indicateurs binaires permettent de comparer deux surfaces de compromis entre elles. La mesure la plus utilisée est la métrique de couverture d'ensemble, appelée métrique $\mathcal{C}$ (ZDT00). La métrique $\mathcal{C}$ évalue la proportion de solutions d'une surface de compromis B qui sont dominées par les solutions d'une surface de compromis A. La formule mathématique permettant de calculer cette métrique $\mathcal{C}$

est la suivante :

$$\mathcal{C}(A,B) = \frac{|\{b \in B, \exists\, a \in A \,:\, a \preceq b\}|}{|B|} \tag{1.6}$$

Cette métrique est comprise entre 0 et 1. Lorsqu'elle est égale à 1, toutes les solutions de B sont dominées ou égales aux solutions de A. A l'inverse, lorsque la métrique $\mathcal{C}$ est nulle aucune des solutions de B n'est dominée par les solutions de A.

Chacun des indicateurs, présentés précédemment, exprime un aspect particulier (forme, qualité, diversité...) de la surface de compromis. Il convient donc d'utiliser plusieurs indices pour avoir une bonne idée globale du comportement de l'algorithme d'optimisation. Tous ces indicateurs sont calculés dans l'espace des objectifs. Aussi, il est possible d'exporter certaines mesures de diversité à l'espace des variables, notamment pour connaître la diversité génotypique de la surface de compromis. Dans le chapitre 4 de ce manuscrit, la notion de « variante » est définie afin de caractériser la diversité génotypique des solutions d'agencement proposées par différentes stratégies d'optimisation.

1.3 Optimisation d'agencement d'espace

Cette troisième partie est dédiée aux problèmes d'agencement d'espace. L'agencement d'espace est une problématique industrielle qui met en œuvre l'optimisation de conception. Après une brève description de cette problématique et des applications industrielles associées, une attention particulière est portée sur les approches et méthodes permettant la description, la formulation et la résolution de ces problèmes.

1.3.1 Définition générale et applications

L'agencement d'espace est habituellement défini comme un problème d'optimisation et différentes définitions de l'optimisation d'agencement sont possibles (CSY02; YFG08). L'idée principale reste cependant toujours la même : *étant donné un ensemble de* ***composants*** *et un* ***contenant****, l'optimisation d'agencement consiste à trouver l'ensemble des* ***variables de positionnement*** *des composants afin de minimiser certains* ***objectifs****, tout en respectant certaines* ***contraintes***. Cette définition générale s'adapte à toutes les applications industrielles. Par exemple Drira *et al.* (DPHG07) et Wäscher *et al.* (WHS07) ont ajusté cette définition à leurs domaines de recherche respectifs : l'agencement de machines dans une usine et les problèmes de découpe et de conditionnement.

Un problème d'optimisation d'agencement peut être défini comme un ensemble d'éléments connectés entre eux et permettant la génération de solutions répondant à un besoin formulé par le concepteur. Ces éléments peuvent donc être assemblés dans une structure commune, illustrée sur la figure 1.9.

L'optimisation d'agencement d'espace trouve de nombreuses applications dans divers domaines in-

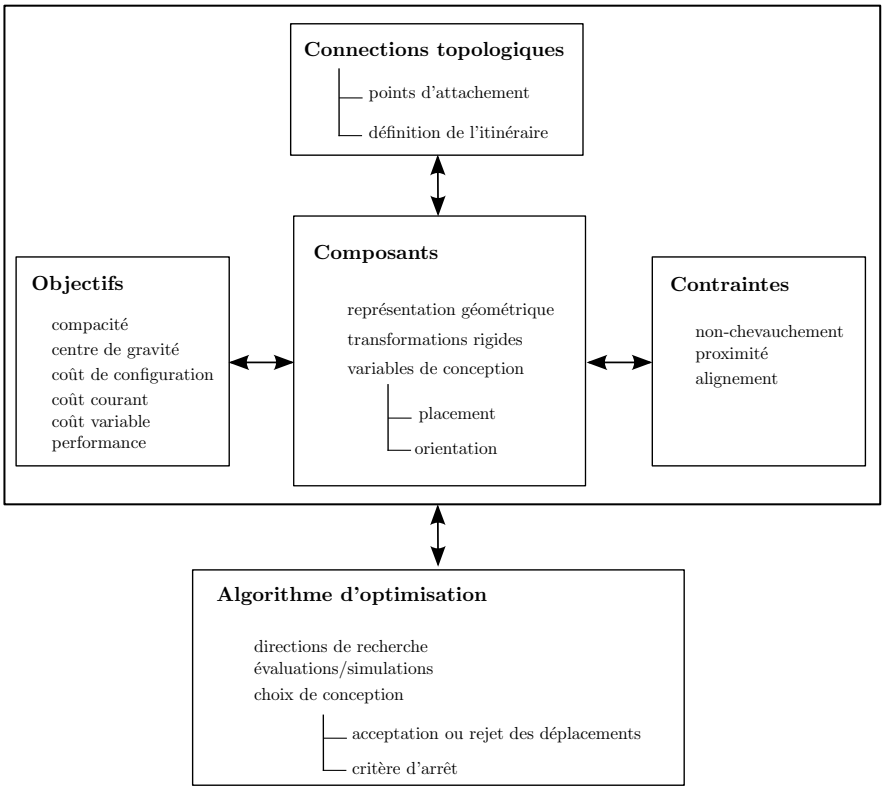
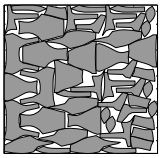


FIGURE 1.9 – *Structure d'un problème d'agencement d'espace, d'après (CSY02)*

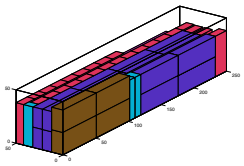
dustriels, tels ceux illustrés sur la figure 1.10. Par exemple, les problèmes de découpe de formes irrégulières sont présents dans les applications industrielles liées à la découpe de matière première (tôle, tissu...) où l'objectif principal est de limiter les coûts de découpe et les chutes générées lors de ces opérations. Les problèmes de chargement et de conditionnement ont pour objectif de placer un certain nombre de composants dans un contenant, en essayant dans la plupart des cas de maximiser le chargement tout en respectant les contraintes de répartition des composants (équilibre des masses, ordre de déchargement...). Il est possible également de citer les problèmes de placement de machines dans une usine, où l'objectif est d'augmenter la productivité en limitant les flux de matériels et d'opérateurs entre les équipements de production.

1.3.2 Description et formulation des problèmes

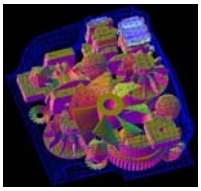
La description et la formulation des problèmes sont deux étapes importantes du processus d'optimisation d'agencement d'espace. Ces deux étapes consistent principalement à créer un modèle géométrique du problème et à traduire les exigences du concepteur en variables d'optimisation, contraintes et objectifs. Les deux sous-sections suivantes montrent les différences qui peuvent exister dans la description et la formulation des problèmes.



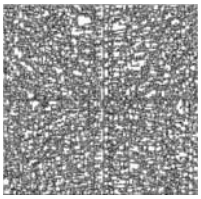
Problème de découpe irrégulière (BS07)



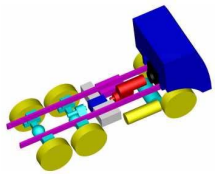
Chargement de container (Pis02)



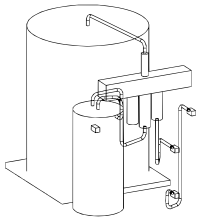
Conditionnement 3D (TFF08)



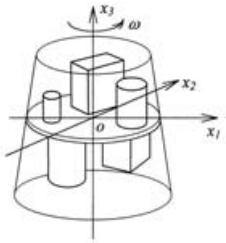
Placement de composants électroniques (Ege03)



Agencement d'un camion (Yin00; YFG08)



Agencement d'une pompe à chaleur (CCD+96)



Agencement d'un satellite (TSLL01; ZTS08)

FIGURE 1.10 – *Applications industrielles des problèmes d'optimisation d'agencement d'espace. Image empruntée à (Jac10)*

1.3.2.1 Variables d'optimisation

Les variables d'optimisation d'un problème d'agencement d'espace définissent généralement les variables de positionnement des composants dans le contenant. Ces variables peuvent être de différents types : continues, discrètes, binaires ou permutations. Le choix du type de variable pour représenter la position et l'orientation des composants dépend de la géométrie des composants, des contraintes et des objectifs du problème.

L'utilisation des variables continues est la représentation du placement de composants la plus naturelle. Le placement des composants à l'aide de ces variables peut être absolu ou relatif lorsque la position d'un composant dépend du positionnement d'un autre composant. Le placement relatif de composants est notamment adapté aux contraintes topologiques présentes dans un problème d'agencement (alignement de composants par exemple). Cependant, les variables continues ne sont pas particulièrement adaptées aux problèmes présentant certaines spécificités (orientation discrète, présence ou non d'un composant...).

Les variables discrètes ou binaires peuvent être utilisées pour positionner les composants sur une grille, notamment lorsque ceux-ci ont une forme rectangulaire ou parallélépipédique. Cette représentation permet de vérifier simplement les contraintes de placement du problème. Ces variables d'optimisation sont également utilisées pour définir l'orientation des composants ou la présence ou

non d'un composant dans l'agencement, notamment pour les problèmes de découpe et de conditionnement.

Les variables de permutation définissent une liste ordonnée d'entiers qui peut représenter la liste des identifiants des composants. Le problème d'agencement est alors un problème d'optimisation combinatoire, où le nombre de solutions est fini mais très grand. Les variables de permutation sont généralement utilisées pour représenter l'ordre d'introduction des composants dans un contenant, comme pour les problèmes de découpe et de conditionnement (Jak96).

1.3.2.2 Contraintes et objectifs

Les contraintes et les objectifs d'un problème d'optimisation d'agencement traduisent les exigences du concepteur. Les contraintes permettent notamment d'identifier les solutions admissibles, c'est-à-dire les solutions qui respectent les contraintes du problème. Dans tous les problèmes d'agencement, les contraintes de non-chevauchement entre composants et les contraintes d'appartenance au contenant sont présentes. Pour les problèmes d'agencement en deux dimensions, le calcul des contraintes de non-chevauchement utilise le plus souvent le calcul du polygone de non-recouvrement (*No-Fit Polygon* en anglais). Le terme de « polygone de non-recouvrement » a été introduit par Adamowicz et Albano en 1976 (AA76) alors que ces derniers s'attachaient à résoudre un problème de découpe de pièces de forme irrégulière. La définition du polygone de non-recouvrement est la suivante (BHKW07) : étant donné deux polygones A et B, le polygone de non-recouvrement est une forme géométrique tracée autour de la frontière d'un polygone. Si le polygone A reste fixe, le polygone B se déplace autour du polygone A en s'assurant qu'il touche la frontière de ce dernier sans jamais l'intersecter. De même, afin de tracer le polygone de non-recouvrement, un point de référence, situé sur le polygone en mouvement, est utilisé. Si le polygone A est le polygone qui reste fixe, le polygone de non-recouvrement est noté : NFP_{AB}. La figure 1.11 illustre la construction du polygone NFP_{AB} pour deux polygones A et B.

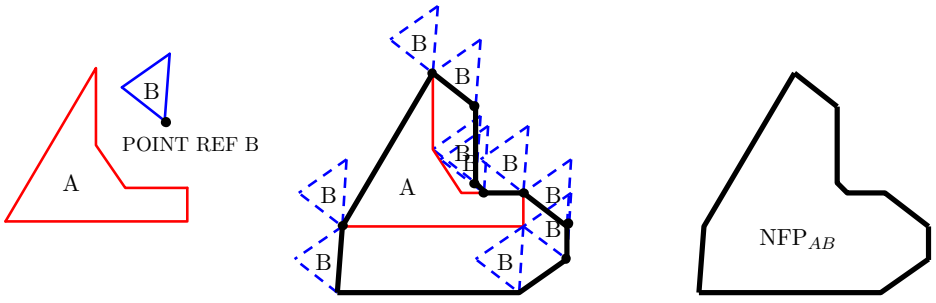


FIGURE 1.11 – *Construction du polygone de non-recouvrement, d'après (BHKW07)*

En résumé, pour savoir si le polygone B chevauche le polygone A, on construit dans un premier temps le polygone de non-recouvrement entre A et B avec un point de référence sur B. Ensuite, si le polygone B est positionné tel que le point de référence est à l'intérieur du polygone de non-

recouvrement NFP_{AB} alors le polygone B chevauche le polygone A.

Bien que le polygone de non-recouvrement soit un excellent outil pour détecter le chevauchement entre deux polygones, il n'est pas beaucoup utilisé dans les applications industrielles d'agencement, notamment pour les problèmes de conditionnement de composants. Ceci est principalement dû à la difficulté d'implémentation de la méthode de construction du polygone de non-recouvrement et au manque d'algorithmes robustes pour la réaliser. Ainsi, dans certains cas d'application, d'autres approches géométriques classiques sont préférées. Par exemple, il est possible de calculer l'aire ou le volume d'intersection entre composants, lorsque ceux-ci ont une forme géométrique relativement simple. Cependant, la méthode utilisant le polygone de non-recouvrement est un outil de détection du chevauchement plus rapide, car les polygones de non-recouvrement peuvent être générés en début du processus d'optimisation d'agencement et non pas besoin d'être recalculés à chaque génération d'une nouvelle solution, à l'inverse des approches géométriques classiques.

Plusieurs approches existent pour construire le polygone de non-recouvrement. Certaines approches, plus simples à implémenter, sont utilisées lorsque les polygones sont convexes (BHKW07). Une de ces méthodes est décrite dans (DTR06). Pour les problèmes d'agencement en deux dimensions où les composants ont une forme non-convexe, ou pour les problèmes d'agencement en trois dimensions, il est possible d'utiliser le principe des sommes de Minkowski (HYB07).

Dans le chapitre 2 de ce manuscrit, une méthode innovante permettant de mesurer l'accessibilité d'un composant depuis l'entrée du contenant est proposée. Cette méthode est basée sur le principe du polygone d'appartenance (*Inner-Fit Polygon* en anglais) qui est lui même une variante du polygone de non-recouvrement.

D'autres contraintes et objectifs sont également formulés dans les problèmes d'optimisation d'agencement pour traduire les exigences du concepteur et les relations qui peuvent exister entre les composants : distance entre composants à respecter, répartition des masses... Ces contraintes et ces objectifs dépendent fortement du cas d'application traité. Par exemple, dans les problèmes d'agencement d'équipements dans une usine, plusieurs contraintes ou objectifs sont formulés pour assurer le bon fonctionnement du processus de production présent dans l'usine.

1.3.3 Classification des problèmes

Il est difficile de trouver dans la littérature scientifique, une typologie complète et générique des problèmes d'agencement d'espace. L'idée d'une classification permettrait de trier les problèmes d'agencement en fonction de la description et la formulation des problèmes, c'est-à-dire la forme des composants, les variables d'optimisation utilisées, les contraintes et les objectifs du problème. Ce classement des problèmes d'agencement permettrait également d'orienter le concepteur vers la stratégie d'optimisation la plus adaptée à son problème.

Il existe cependant certaines classifications proposées pour des problèmes d'agencement particuliers. Par exemple, Harald Dyckhoff a été le premier à proposer une typologie complète des problèmes de

découpe et de conditionnement (*cutting and packing problems* en anglais) (Dyc90). La typologie de Dyckhoff est basée sur quatre paramètres qui permettent d'identifier 96 problèmes. Cette typologie n'a pas connu une reconnaissance très importante au niveau international. En effet, les scientifiques travaillant sur ces problèmes se sont aperçus par exemple que certains problèmes de découpe et de conditionnement pouvaient être classés de plusieurs façons différentes via cette typologie. Ils ont également émis quelques réserves concernant le niveau de détail de ce classement car certains problèmes se retrouvaient dans la même catégorie alors que leurs approches de résolution étaient totalement différentes. Par conséquent, Wäscher *et al.* (WHS07) ont apporté quelques modifications à cette typologie afin de l'améliorer encore. Cette typologie est illustrée par la figure 1.12.

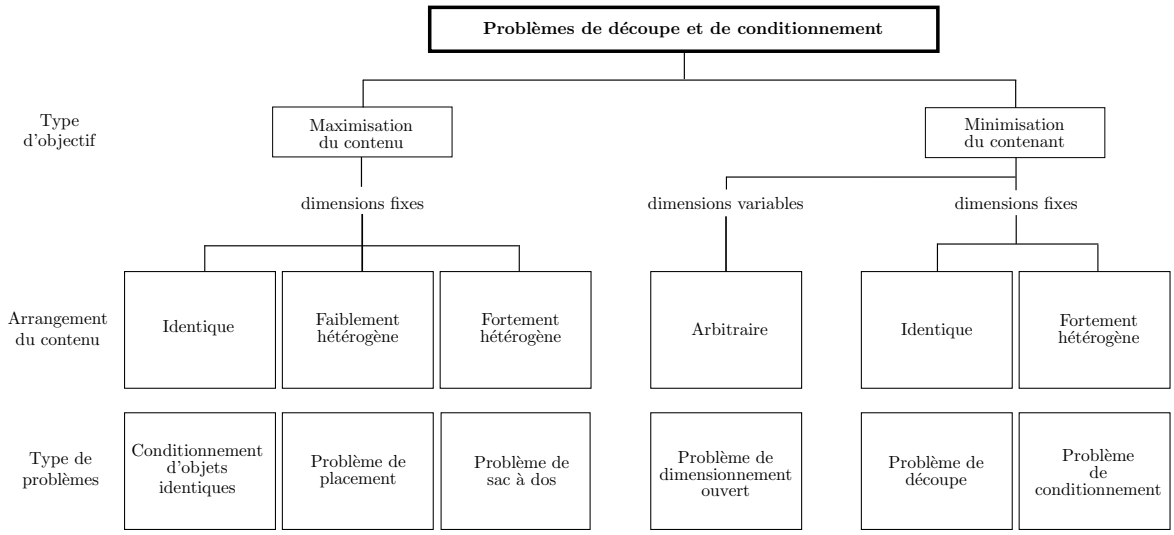


FIGURE 1.12 – *Typologie des problèmes de découpe et de conditionnement, d'après (WHS07)*

La classification proposée sur la figure 1.12 distingue cinq critères de tri des problèmes de découpe et de conditionnement qui correspondent aux quatre critères de Dyckhoff plus un critère portant sur la forme des composants. Les critères définis précédemment par Dyckhoff sont modifiés afin de pallier les insuffisances de la typologie initiale. En effet, le type d'affectation correspond maintenant à deux types de problèmes : ceux qui visent à minimiser la taille du contenant et ceux qui visent à maximiser la taille du contenu, c'est-à-dire le nombre de composants. De même, les critères portant sur la nature des contenants et des composants sont redéfinis par trois valeurs : identique, faiblement hétérogène et fortement hétérogène. Les critères portant sur la dimension du problème, la nature des contenants et la forme des composants du contenu ne sont pas pris en compte.

Un aperçu global de ces problèmes de découpe et de conditionnement est présenté dans (DD92). On y trouve notamment les problèmes de découpe (GG63; BKW04), les problèmes de sac à dos (KPP04; EP09)ou encore les problèmes de chargement de palettes (BMN05).

1.3.4 Stratégies de résolution

1.3.4.1 Présentation générale des stratégies d'optimisation

Dans les problèmes d'optimisation d'agencement d'espace traités dans la littérature scientifique, de nombreux algorithmes d'optimisation sont utilisés (CSY02). En effet, dans la plupart des applications, les stratégies de résolution sont dédiées à un problème d'agencement particulier. Ainsi, ils sont adaptés aux spécificités de ce problème et sont très efficaces dans la recherche de solutions optimales. Cependant, il est difficile d'adapter ces stratégies à un grand nombre de problèmes d'agencement différents. Actuellement, les recherches s'orientent donc vers le développement de nouvelles stratégies d'optimisation, plus génériques qui permettent de résoudre un grand nombre de problèmes différents avec des temps de calcul raisonnables.

Les stratégies d'optimisation utilisées pour résoudre les problèmes d'agencement d'espace peuvent être divisées en plusieurs catégories. Par exemple, on distingue les approches mono-objectif et multiobjectif, les approches globales (qui recherchent dans tout l'espace de conception) et les approches locales (qui recherchent au voisinage d'une solution initiale). Aussi, il est possible de classer les algorithmes d'optimisation en deux groupes :

- les méthodes « exactes », qui garantissent le respect des contraintes de conception formulées par le concepteur, tout au long du processus d'optimisation. Parmi ces méthodes, nous pouvons citer les algorithmes par évaluation et séparation (*branch and bound* en anglais) (ST95), les méthodes de programmation linéaire (Bea85) ou encore les algorithmes basés sur le calcul du gradient (LB94). Une autre approche exacte est basée sur les techniques de programmation par contraintes. Un exemple de cette stratégie, appliquée au problème de placement optimal de locaux, est décrit dans (Med96),
- les méthodes « approchées », qui peuvent proposer des solutions non nécessairement optimales et ne garantissant pas le respect des contraintes du problème d'agencement. Parmi ces méthodes, nous pouvons citer par exemple, l'algorithme du recuit simulé (Sec98; SC95; SC97), la méthode de recherche par motif (SC00) et l'algorithme génétique (YFG08; SLGL00).

Le choix d'une stratégie d'optimisation est fortement lié à la formulation du problème d'optimisation d'agencement. Aussi, il est possible de combiner plusieurs méthodes et de mettre au point un algorithme d'optimisation « hybride ». Le paragraphe suivante décrit une stratégie d'optimisation hybride, basée sur l'utilisation d'un algorithme génétique.

1.3.4.2 Algorithme génétique hybride

Cette partie s'intéresse à une stratégie de résolution hybride pour l'optimisation d'agencement d'espace, basée sur le couplage entre un algorithme génétique et un algorithme de séparation. Cette méthode d'optimisation a été développée par G. Jacquenot (Jac10; JBMW09). Cet algorithme hybride permet de traiter une grande variété de problèmes d'optimisation d'agencement multiobjectif

et s'adapte aux problèmes où les composants ont des géométries complexes.

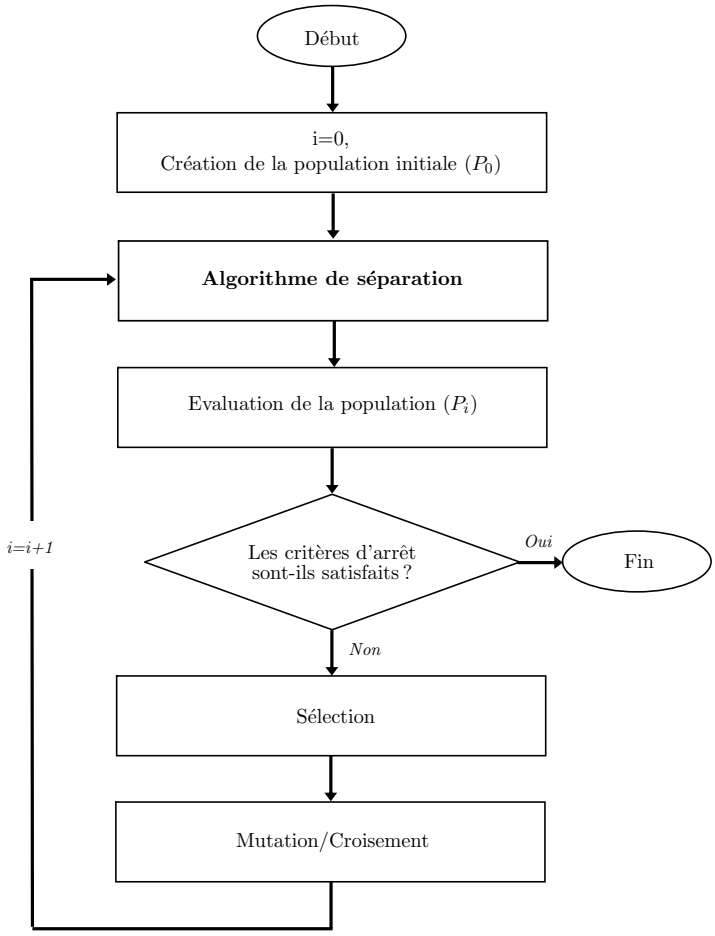


FIGURE 1.13 – *Principe de fonctionnement de l'algorithme hybride proposé par (Jac10)*

Le principe de fonctionnement de cet algorithme hybride est illustré sur la figure 1.13. La structure de cet algorithme est très proche de celle de l'algorithme génétique. Un algorithme de séparation est intégré à l'algorithme génétique et modifie les variables de positionnement des composants afin que l'agencement respecte toutes les contraintes de conception. En résumé, avant d'évaluer un individu, l'algorithme hybride teste si l'individu respecte les contraintes de conception. Si ces contraintes sont respectées, les objectifs sont évalués et l'algorithme poursuit son processus de résolution en faisant évoluer cet individu vers un individu meilleur, via les différents opérateurs génétiques. A l'inverse, si les contraintes de conception ne sont pas respectées, l'algorithme de séparation est appliqué et l'algorithme poursuit ensuite son processus de résolution. Plus d'informations sur cet algorithme de séparation sont disponibles dans (Jac10) et dans le chapitre 3 faisant référence à la stratégie modulaire. Finalement, cet algorithme est la combinaison entre un algorithme d'optimisation globale (algorithme génétique) et un algorithme d'optimisation locale (algorithme de séparation).

1.4 Optimisation interactive

Cette partie vise à présenter différentes méthodes et outils qui permettent l'interaction entre le concepteur et le processus d'optimisation, notamment pour les problèmes d'agencement d'espace. Tout d'abord, nous examinerons quel rôle peut jouer le concepteur dans le processus d'optimisation. Nous détaillerons ensuite quelques concepts permettant d'intégrer le concepteur dans les différentes étapes du processus d'optimisation. Nous aborderons également la notion de réalité virtuelle qui englobe un ensemble d'outils facilitant l'interaction avec le concepteur.

1.4.1 Rôle du concepteur en optimisation

Même si de nombreuses méthodes efficaces ont été développées afin de résoudre le plus grand nombre de problèmes d'optimisation, le concepteur joue toujours un rôle important dans la résolution de ces problèmes. En effet, le traitement de ces problèmes ne peut pas se faire uniquement de manière automatique, c'est-à-dire « la machine ne remplace pas l'homme ». Cette nécessité d'une complémentarité entre le concepteur et l'ordinateur s'explique par la difficulté croissante des problèmes d'optimisation qui dépend de plusieurs critères (dNE05) :

- la formulation du problème fait intervenir des critères ou des connaissances subjectives et perceptives, qui varient d'un concepteur à l'autre,
- certains paramètres du problème ne sont pas totalement maîtrisés et doivent subir des ajustements en cours d'optimisation,
- les problèmes sont souvent multicontrainte et multiobjectif et il faut donc faire un compromis basé sur des préférences du concepteur,
- les applications industrielles diffèrent des problèmes testés et décrits dans la littérature scientifique et les méthodes utilisées doivent être adaptées,
- les ressources informatiques peuvent être insuffisantes pour résoudre les problèmes les plus complexes.

Les problèmes d'optimisation d'agencement d'espace font partie de cette classe de problèmes d'optimisation difficiles. L'interaction avec le concepteur est donc nécessaire. Cette interactivité peut se matérialiser tout au long du processus d'optimisation, c'est-à-dire dans les trois grandes étapes définies par :

1. **la formulation du problème** : l'expertise du concepteur, ses connaissances métier, subjectives et perceptives aident à la définition des contraintes et des objectifs du problème d'optimisation,
2. **l'algorithme d'optimisation** : intégrer les préférences du concepteur dans la résolution du problème d'optimisation et améliorer les performances de l'algorithme,
3. **la prise de décision** : les choix de conception appartiennent toujours au concepteur.

1.4.2 Interactivité dans le processus d'optimisation

Cette section décrit un certain nombre de méthodes qui permettent d'intégrer le concepteur dans les trois grandes étapes du processus d'optimisation.

1.4.2.1 Formulation du problème

La formulation du problème d'optimisation consiste principalement à traduire le besoin du concepteur en variables, contraintes et critères d'optimisation. Différentes approches existent pour capter le besoin et les perceptions du concepteur. Par exemple, la méthode QFD (*Quality Function Deployment*) (Aka00) traduit les besoins du concepteur et les relie avec les différentes étapes de la vie d'un produit. La méthode QFD aide le concepteur à identifier explicitement ses besoins et à les corréler aux caractéristiques techniques du produit ou du système. La représentation la plus caractéristique du QFD est la maison de la qualité (*the House of Quality*). Aussi, les méthodes véhiculées par le « *Kansei engineering* », visent à étudier et relier les perceptions et sensations du concepteur à des critères physiques et des variables de conception (Nag95; YP04; Poi05; YP02). Les méthodes d'analyse sensorielle, initialement développées pour les applications relatives aux produits alimentaires (APR65; SSO+74), consistent à faire participer le concepteur à différents tests pour fournir une réponse perceptive. Parmi ces méthodes, nous pouvons citer par exemple l'épreuve de notation et de cotation. Enfin, la théorie de l'utilité multiattribut est aujourd'hui une approche largement utilisée dans les applications d'ingénierie pour aider le concepteur à formuler un critère d'optimisation basé sur ses préférences et ses perceptions (LCS06).

Pour certaines applications où il est difficile de formuler du premier coup toutes les contraintes et tous les objectifs du problème d'optimisation, il est possible d'adopter une formulation dynamique du problème via l'interaction entre l'algorithme d'optimisation et le concepteur. Ce concept est abordé dans la section suivante.

1.4.2.2 Algorithme d'optimisation

Le concepteur peut intervenir dans l'algorithme d'optimisation afin d'améliorer les performances de ce dernier. Par exemple, il peut proposer à l'algorithme une solution (ou un ensemble de solutions) initiale faisable avec des performances correctes. L'algorithme d'optimisation va partir de cette solution initiale pour améliorer les critères d'optimisation. Le temps de calcul du processus d'optimisation est réduit car la solution initiale proposée est à priori de meilleure qualité qu'une solution aléatoirement proposée.

L'interaction entre le concepteur et l'algorithme peut se faire dans le cadre d'une reformulation dynamique du problème d'optimisation. Michalek *et al.*(MP02) proposent une méthode d'optimisation interactive dédiée aux problèmes d'architecture. La méthode permet tout d'abord au concepteur de proposer à l'algorithme une solution initiale. Ensuite l'algorithme optimise la solution en prenant en

compte les contraintes et les objectifs définis par le concepteur. Le concepteur visualise et analyse la solution optimisée, modifie la solution proposée, ajoute ou enlève des variables, des contraintes ou des objectifs et relance l'optimisation à partir de la solution générée.

Aider l'algorithme dans sa recherche de solutions optimales peut également se faire de manière interactive via l'utilisation des systèmes multiagent qui permettent la collaboration entre le concepteur et d'autres « agents » qui participent indépendamment à la résolution du problème (CDY98).

D'autres algorithmes interactifs forment une classe de méthodes utilisées pour la résolution des problèmes d'optimisation multiobjectif (CS02). L'interactivité permet ici au concepteur d'exprimer ses préférences sur les différents critères d'optimisation qui sont définis. Parmi ces méthodes qui sont détaillées dans (Mie99), nous pouvons citer la méthode du compromis par substitution (*surrogate worth tradeoff* en anglais, SWT) (HHF75; Sak78), la méthode GDF (Geoffrion-Dyer-Feinberg) (GDF72), la méthode de Tchebycheff (SC83) ou encore la méthode NIMBUS (MM95). Ces méthodes interactives sont basées sur une structure itérative divisée en plusieurs phases : la phase d'initialisation (solution initiale proposée aléatoirement par l'ordinateur), la phase de dialogue (dialogue avec le concepteur pour connaître ses préférences) et une phase de calcul (génération de nouvelles solutions par l'ordinateur).

Il est possible également d'identifier une autre classe de méthodes interactives dédiées à la prise en compte, dans l'algorithme, des perceptions du concepteur, des critères subjectifs et autres aspects esthétiques (son, image...). Ces méthodes traitent ces critères qualitatifs de manière mono-objectif ou multiobjectif (PCWB00; SKK99; OTR03; KTA04). Parmi ces méthodes, nous pouvons citer par exemple l'algorithme génétique interactif (*Interactive Genetic Algorithm* en anglais, IGA). Dans le chapitre 4 de ce manuscrit, un IGA est proposé pour une exploration perceptive de solutions d'agencement. Ces algorithmes génétiques interactifs ont une structure similaire aux algorithmes génétiques traditionnels mais font intervenir l'utilisateur dans la phase d'évaluation des solutions. Brintrup *et al.* (BRT07) proposent deux IGA pour associer les critères quantitatifs et qualitatifs dans le processus d'optimisation de conception : l'algorithme génétique séquentiel et l'algorithme génétique multiobjectif.

L'IGA séquentiel sépare dans le temps le traitement des critères qualitatifs et le traitement des critères quantitatifs. Après avoir été créée aléatoirement, la population initiale d'individus est affichée et présentée au concepteur. Le concepteur a alors le choix de procéder à l'optimisation des critères qualitatifs ou des critères quantitatifs. Si le choix porte sur les critères qualitatifs, le concepteur doit évaluer qualitativement chaque individu en le notant par exemple de 0 (excellent *design*) à 9 (très mauvais *design*). L'évaluation manuelle des *designs* apparaît, pour l'algorithme génétique, comme une évaluation traditionnelle d'une population d'individus. Ensuite, l'IGA génère une nouvelle population de *designs* qui est de nouveau présenté au concepteur qui l'évalue qualitativement. Ce processus est réitéré jusqu'à ce que les critères d'arrêts définis par le concepteur soient respectés. Ensuite, l'IGA reprend le même processus mais en considérant maintenant les critères quantitatifs, qui sont calculés automatiquement.

L'IGA multiobjectif utilisé dans (BRT07) est une version modifiée du NSGA-II, initialement proposé par (DPAM02). De la même manière que l'IGA séquentiel, une population initiale, aléatoirement créée, est présentée au concepteur. Le concepteur évalue qualitativement chaque individu en le notant. Les critères quantitatifs sont également calculés à ce stade de l'algorithme. Ensuite, l'IGA génère une nouvelle population de solutions, en considérant au même niveau l'évaluation des critères quantitatifs et qualitatifs. L'IGA séquentiel et l'IGA multiobjectif ont été testés sur un cas d'application d'optimisation d'agencement d'espace dans une usine (BRT07). L'objectif de ce problème d'optimisation multiobjectif est de chercher les dimensions optimales des différentes pièces de l'usine afin de minimiser le coût de construction de l'usine tout en satisfaisant le jugement personnel des différents experts travaillant sur le projet. Pour chaque IGA, la population d'individus est composée de 12 *designs*. Le nombre maximal d'itérations est fixé à 10, ce qui signifie que le concepteur est susceptible d'évaluer qualitativement au maximum 120 solutions.

Une autre version de l'IGA est proposé par Kelly *et al.* dans (KPS08). Dans cette version de l'IGA, l'optimisation se fait uniquement sur des critères qualitatifs. L'intégration des perceptions du concepteur n'est pas effectuée par l'évaluation manuelle de chaque individu mais pour chaque population d'individus. En effet, pour chaque population, le concepteur choisit une ou plusieurs solutions correspondant le plus à ses critères perceptifs. Via un mécanisme de sélection par roulette, les individus sélectionnés par le concepteur ont plus de probabilité d'être utilisés par les opérateurs génétiques (croisement, mutation et sélection) afin de générer une nouvelle population de solutions. La figure 1.14 illustre le fonctionnement de cet IGA. Poirson *et al.* (PPB+11) présentent une étude détaillé de cet IGA, appliquée à l'évaluation perceptive d'un verre à pied.

Nous pouvons trouver dans (CYD10) une version semblable de l'IGA, appliquée au dessin du profil d'une voiture. L'IGA qui est proposé est également basé sur des critères uniquement qualitatifs mais le concepteur évalue chaque individu qui lui est proposé. Il donne une note, comprise entre deux maxima, permettant de quantifier son jugement perceptif de la solution proposée.

1.4.2.3 Prise de décision

La prise de décision, faite par le concepteur, est une étape importante du processus de résolution d'un problème d'optimisation multiobjectif. Cette étape consiste, pour le concepteur, à faire un choix de conception parmi les solutions optimales proposées par l'algorithme d'optimisation. Ce choix de conception se fait par l'intégration de l'expertise et du jugement personnel du concepteur dans l'exploration d'un ensemble de solutions optimales dans un espace multidimensionnel. Les méthodes et outils permettant cette prise de décision peuvent être regroupés dans le concept de « *design by shopping* » initialement proposé par Balling (Bal99).

Cette prise de décision nécessite un ensemble d'outils graphiques interactifs permettant l'exploration de solutions. La difficulté réside ici dans la représentation des données dans des espaces multidimensionnels qui ne sont pas seulement limités à la visualisation en trois dimensions. L'idée principale est

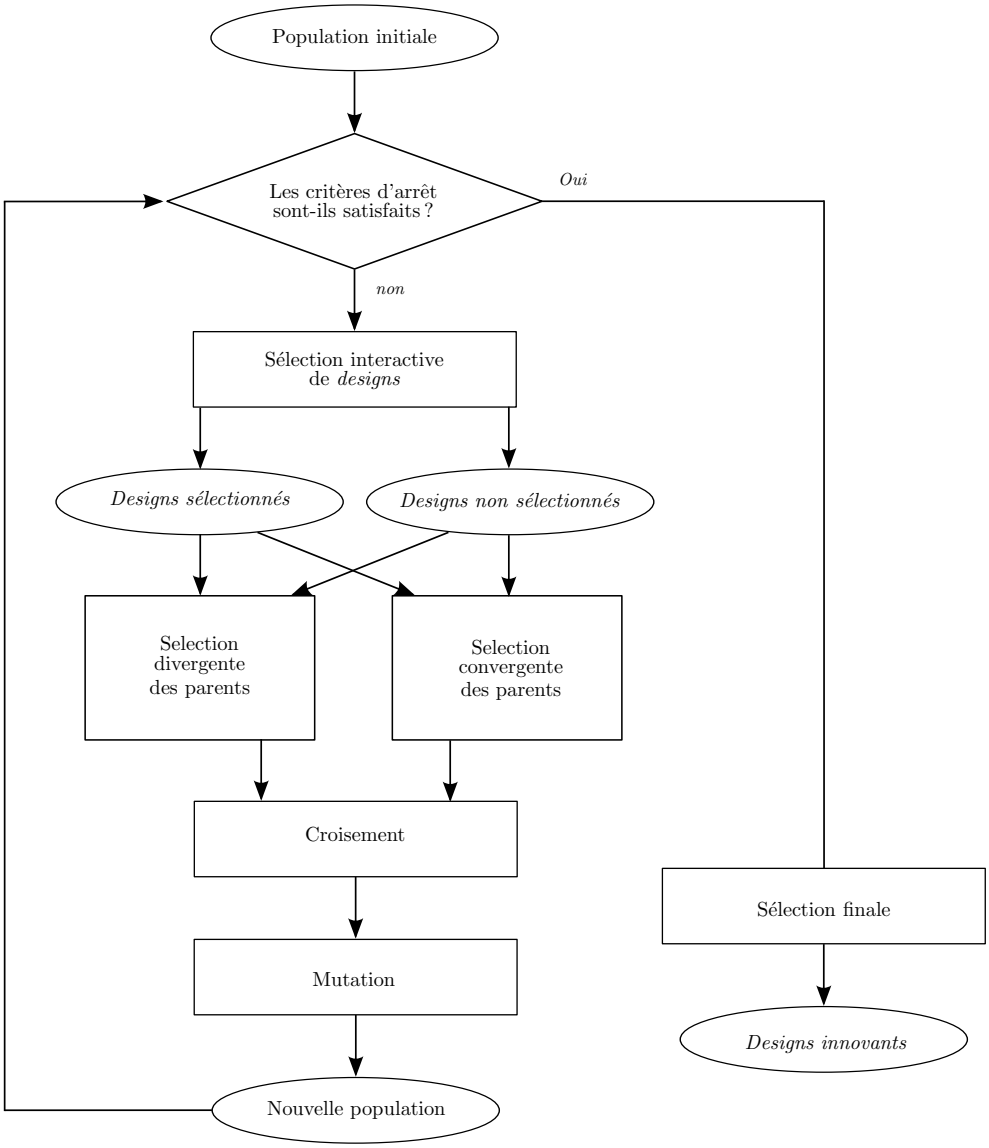


FIGURE 1.14 – *Algorithme génétique interactif (IGA). Image empruntée à (KPS08)*

donc de permettre au concepteur non seulement de visualiser les données mais également d'interagir avec celles-ci . Plusieurs logiciels libres et commerciaux proposent des interfaces graphiques pour explorer, dans des espaces multidimensionnels, des données issues d'un calcul d'optimisation par exemple. Parmi les outils utilisés, nous pouvons citer par exemple le nuage de points ou encore le graphe parallèle, deux outils qui sont détaillés dans le chapitre 4 de ce manuscrit. Un tableau récapitulatif des logiciels permettant la visualisation multidimensionnelle de données est détaillé dans (SYSH03). Un exemple d'interface graphique d'exploration de données est décrit dans (SYS+04). Cette interface, nommée « *Advanced Trade Space Visualizer* » (ATSV) est utilisée pour l'analyse des différentes solutions de compromis générées pour la conception d'un satellite. La figure 1.15 illustre l'interface graphique du logiciel ATSV.

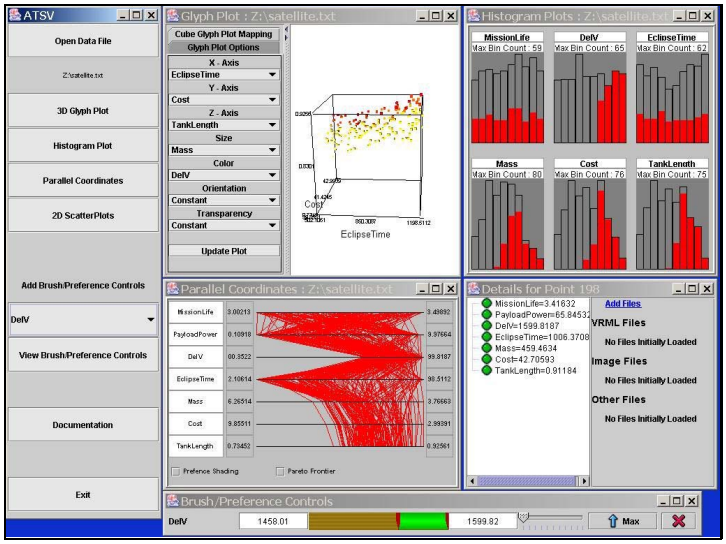


FIGURE 1.15 – *ATSV : exemple d'interface graphique de prise de décision. Image empruntée à (SYS+04)*

D'autres récentes recherches, sur cette problématique de représentation des données en n-dimensions, sont proposées dans (EW07; MM05; ALC+04; EL02). Ces études permettent au concepteur de visualiser et comparer les solutions Pareto-optimales générées par l'algorithme d'optimisation et fournissent ainsi au concepteur des outils graphiques d'aide à la décision.

1.4.3 Outil pour l'optimisation interactive : la réalité virtuelle

Aujourd'hui, la réalité virtuelle révolutionne la façon dont nous voyons, percevons et analysons le monde qui nous entoure. Des balbutiements de la représentation tridimensionnelle au siècle dernier jusqu'aux dernières plate-formes de réalité virtuelle mises au point, le concepteur est transporté dans un monde où les connaissances sont beaucoup plus faciles à acquérir pour l'esprit humain.

La réalité virtuelle est par définition une simulation informatique interactive immersive, visuelle, sonore et/ou haptique, d'environnements réels ou imaginaires. Ce concept, relativement ancien, est depuis une cinquantaine d'années en pleine évolution. Nous pouvons citer par exemple, parmi les précurseurs des techniques de réalité virtuelle, Ivan Sutherland (MIT, puis Université d'Utah) qui a conçu en 1968-1970 l ' « *Ultimate Display* », premier casque de visualisation asservi aux mouvements de la tête. Depuis, les nombreuses recherches dans ce domaine permettent d'intégrer ces outils de réalité virtuelle dans des applications industrielles elles aussi très diverses et très nombreuses. Parmi elles, nous trouvons les applications médicales (traitement des phobies, simulation de chirurgie, traitement des convalescents, mise au point de prothèses orthopédiques), les applications nucléaires (démantèlement d'installations en milieux contaminés), la visualisation scientifique, la recherche fondamentale, la domotique, la conservation du patrimoine culturel... Le développement des outils de réalité virtuelle est également présent dans le domaine universitaire. Les laboratoires les plus

connus dans le monde sont notamment le *Virtual Reality Laboratory* de l'Université de l'Illinois ou encore le *Hit Lab* (*Human Interface Technology Lab*) de l'Université de Washington. En France, les recherches les plus avancées dans ce domaine sont orchestrées par l'INRIA (Institut National de la Recherche et Informatique et en Automatique). Un ouvrage très détaillé regroupe tous les concepts, les outils et les applications liées à la réalité virtuelle (Fuc06). Il est également possible de trouver dans (NF10) de nombreux exemples de recherches récentes sur cette thématique.

Comme le font remarquer Jimeno *et al.* (JP07), les avancées technologiques majeures dans le domaine de la réalité virtuelle se sont opérées ces dix dernières années, et ceci découlant du développement important des capacités informatiques et de la miniaturisation des dispositifs sensoriels. La réalité virtuelle se différencie des autres types d'interface utilisateur par deux facteurs importants :

- la présence : le fait de donner à l'utilisateur l'illusion d'être intégrer parfaitement dans un environnement virtuel. Certains scientifiques distinguent la notion d'immersion de la notion de présence bien que ces deux concepts soient très liés,
- l'interaction : le fait de donner à l'utilisateur la possibilité de modifier l'environnement virtuel dans lequel il se trouve immergé. L'utilisateur peut interagir avec ce monde virtuel grâce à des dispositifs physiques (boutons, joysticks...), des dispositifs virtuels ou encore directement via les mouvements de l'utilisateur (direction des yeux, suivi des gestes...).

La figure 1.16 illustre quatre exemples d'interfaces de la réalité virtuelle qui permettent l'interaction entre l'utilisateur et le produit. La plus simple et la plus utilisée des interfaces est la souris d'ordinateur. Une interface un peu plus évoluée est le bras haptique permettant la manipulation d'objets en trois dimensions. Des retours d'efforts sont possibles avec cette interface haptique. Les deux dernières illustrations sont des interfaces visuelles permettant l'immersion de l'utilisateur dans son système : le casque stéréoscopique et la salle immersive.

FIGURE 1.16 – *Exemples d'interface homme/machine pour la réalité virtuelle*

Les outils de réalité virtuelle peuvent donc être utilisés pour les problématiques d'optimisation d'agencement d'espace. En effet, ces outils ont de nombreux avantages pour ces applications :

- la visualisation directe de solutions proposées par une stratégie d'optimisation, via par exemple un affichage interactif simple sur ordinateur ou un affichage à l'échelle 1 dans une salle immersive. Cette visualisation permet d'afficher de nombreux agencements sans avoir besoin de les prototyper physiquement,
- la synchronisation du prototype virtuel avec les différentes phases du processus d'optimisation,
- la collaboration entre les experts du problème d'optimisation d'agencement, même si ces derniers

ne sont pas physiquement au même endroit,
- l'immersion du concepteur ou de l'utilisateur dans l'agencement. Via des interfaces interactives, l'utilisateur peut interagir avec une solution et exprimer de manière efficace son jugement personnel sur celle-ci.

Un exemple de plate-forme d'optimisation d'agencement basée sur l'utilisation d'outils de la réalité virtuelle est proposé dans (ZhYf03). Les différents éléments qui composent cette plate-forme sont illustrés sur la figure 1.17 et sont énoncés ici :
- l'interface utilisateur : permet au concepteur de communiquer avec son environnement virtuel. Par cette interface, le concepteur choisit les objets à placer parmi un catalogue d'éléments et peut les placer dans son espace virtuel. Cette interface permet également au concepteur d'être immergé dans l'environnement virtuel (écran large, gants munis de capteurs...) et ainsi mieux apprécier la qualité de la solution créée,
- l'espace virtuel : se compose de l'environnement virtuel (textures, lumière...), des objets virtuels à positionner et de la détection de collisions. Dans cet espace virtuel, le concepteur peut déplacer les objets, les enlever et grâce aux retours sensoriels, détecter les problèmes d'agencement (collisions entre composants par exemple),
- l'algorithme d'optimisation : stratégie de résolution choisie par le concepteur suivant la formulation de son problème d'agencement (variables, contraintes et objectifs),
- le système expert : système qui guide le processus d'optimisation et aide les experts à prendre leur décisions. Il se compose généralement d'une base de connaissances, d'un système de gestion des informations et d'une interface utilisateur,
- les modèles CAO et RV : les objets à positionner sont directement issus des modèles CAO dessinés par le concepteur. Ensuite, il faut convertir ces modèles en modèles adaptés à la représentation en environnement de réalité virtuelle,
- la simulation du procédé : une fois l'agencement de l'espace terminé, le concepteur peut simuler un procédé industriel lié à cet agencement. Par exemple, dans une usine, le processus de fabrication peut être simulé pour visualiser les flux de matières entre les différentes machines qui ont été précédemment positionnées,
- les sorties : donnent au concepteur un compte-rendu des performances de sa conception (espace utilisé, espace vide, collisions...).

Une des grandes problématiques soulevée par une telle plate-forme est l'association de données graphiques et de données numériques. En effet, dans le cadre d'un problème d'optimisation d'agencement d'espace, la visualisation graphique de l'agencement est très importante, car elle permet d'immerger le concepteur dans l'environnement de conception et de capter de manière efficace son expertise et ses perceptions. Aussi, il est important d'ajouter à cette visualisation un retour sur les performances de la solution optimisée, c'est-à-dire des valeurs numériques sur les contraintes et les objectifs d'optimisation.

Un autre exemple d'application de la réalité virtuelle à la problématique d'agencement d'espace

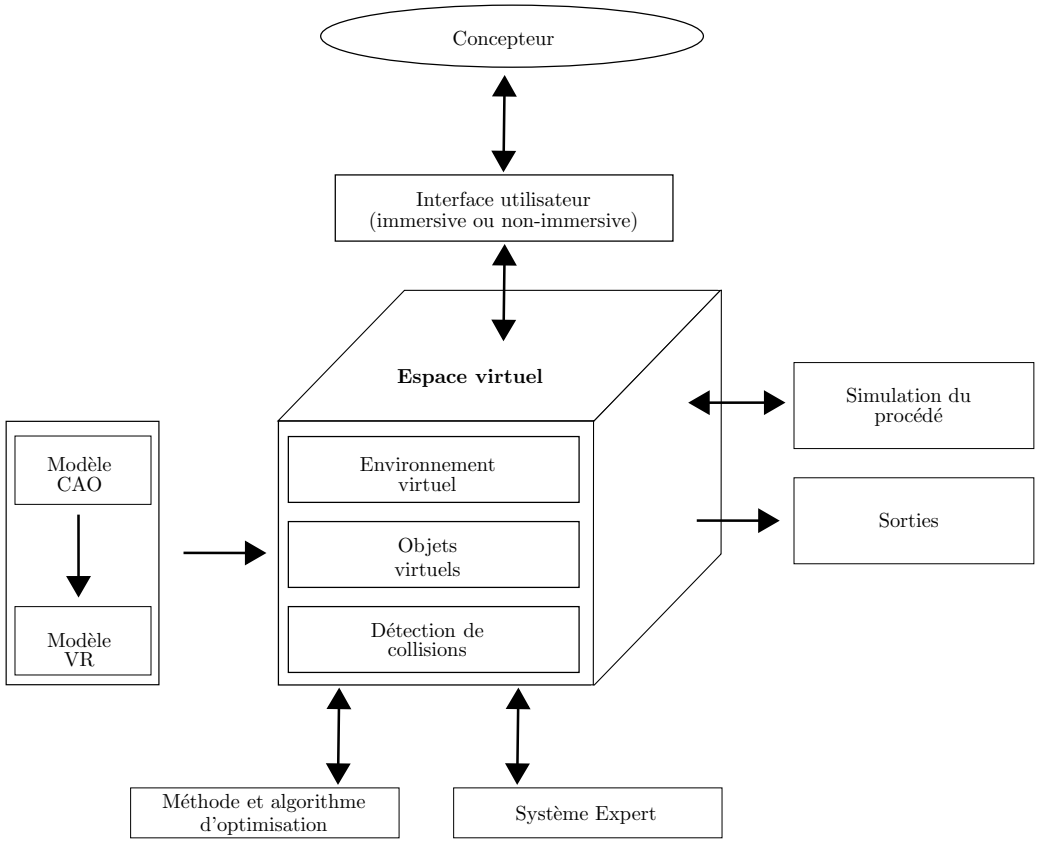


FIGURE 1.17 – *Plate-forme d'agencement basée sur l'utilisation de la réalité virtuelle. Image empruntée à (ZhYf03)*

est proposé dans (KRB10). Dans cette étude, un environnement de réalité virtuelle est associé à un système utilisant les techniques de programmation par contraintes. Le concepteur est immergé dans l'agencement et participe à la recherche de solutions admissibles en aidant à la formulation dynamique du problème.

1.4.4 Limites des stratégies d'optimisation interactives

L'interactivité entre le concepteur et le processus d'optimisation est un concept aujourd'hui largement répandu dans les applications d'ingénierie, notamment dans les problèmes où les perceptions humaines jouent un rôle important sur les performances du produit ou du système. Afin que cette interaction avec le concepteur apporte entière satisfaction, Brintrup *et al.* (BRT07) énumère un certain nombre de limites à l'utilisation des méthodes interactives :

– la fatigue humaine : caractérise l'incapacité du concepteur à évaluer manuellement un grand nombre de solutions. Les algorithmes doivent donc opérer sur des populations restreintes d'individus. Des recherches récentes portent sur la prise en compte de cette fatigue dans le processus d'optimisation interactif (WT05; WWC05),
– la granularité : apparaît lorsque que deux solutions très différentes dans l'espace des critères quan-

titatifs sont deux solutions proches dans l'espace des critères subjectifs. L'évaluation perceptive des solutions est alors plus difficile,
- les contradictions du concepteur : l'évaluation subjective, faite par le concepteur, de deux solutions identiques peut varier au cours du temps. Ceci est principalement dû à la fatigue du concepteur et au fait qu'il ne se rappelle pas nécessairement des *designs* déjà notés,
- les interactions possibles entre critères quantitatifs et qualitatifs : une des questions soulevées par ces méthodes et de savoir s'il faut montrer au concepteur les performances quantitatives d'un *design* en cours d'une évaluation qualitative. L'évaluation du concepteur peut-être influencée par ces données quantifiées,
- la notation fixe des *designs* : au cours du processus d'optimisation, le concepteur peut évaluer qualitativement un individu par une note comprise entre deux extrema. Ces extrema ne varient pas au cours du processus de convergence de l'algorithme et par conséquent, il devient difficile de privilégier une solution par rapport à une autre qui a obtenu une excellente note,
- le manque de vision globale sur la totalité des *designs* : lorsque le concepteur évalue une solution, il peut la comparer à la solution précédente ou aux solutions appartenant à la même population. Il ne peut pas la comparer par rapport à la totalité des *designs* de l'espace de conception.

1.5 Objectifs de recherche

Les travaux de recherche effectués dans ce doctorat proposent une méthode d'optimisation pour résoudre les problèmes d'agencement d'espace. Cette démarche de conception se veut être une approche :
- **intégrée** : l'approche doit résoudre le problème dans sa globalité, depuis l'écoute du besoin du concepteur jusqu'à la prise de décision finale en termes de choix de conception,
- **générique** : l'approche doit pouvoir s'adapter au plus grand nombre de problèmes d'agencement,
- **interactive** : l'approche doit intégrer le concepteur dans la génération d'une ou plusieurs solutions idéales.

Dans un premier temps, il est possible de représenter cette démarche d'optimisation par l'approche systémique. La méthode est vue comme un système de résolution générique des problèmes d'optimisation d'agencement d'espace. Ce système est modélisé par l'analyse fonctionnelle descendante, illustrée sur la figure 1.18.

En entrée de la méthode, il y a le besoin du concepteur. Dans ce besoin, on trouve toujours des composants à placer dans un contenant et un certain nombre d'exigences et de connaissances métier exprimées par le concepteur. Dans la plupart des cas, ce concepteur n'est pas un expert de l'optimisation de conception. Il existe donc dans la démarche, deux étapes qui permettent de traduire ce besoin en un problème d'optimisation. Il faut d'une part créer un modèle géométrique de l'agencement et traduire toutes les exigences du concepteur en variables d'optimisation, contraintes et objectifs. Après ces deux étapes, la méthode suggère au concepteur d'évaluer l'indice de faisabilité

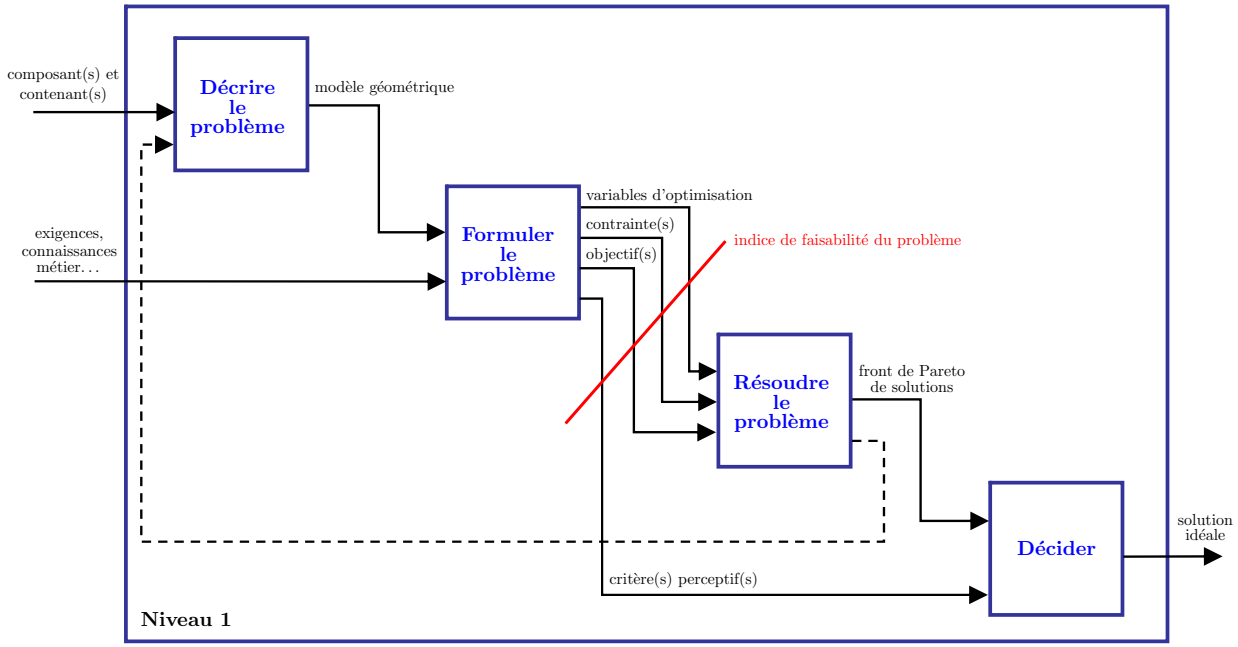


FIGURE 1.18 – *Analyse fonctionnelle descendante de la méthode d'optimisation*

du problème d'optimisation d'agencement afin de savoir s'il faut continuer les étapes suivantes de la méthode.

Ensuite, la méthode repose sur l'utilisation d'une stratégie d'optimisation, la plus générique et efficace possible, qui permet d'obtenir une ou plusieurs solutions optimales au problème d'agencement. Parmi les exigences exprimées par le concepteur, certaines sont difficiles à être intégrées directement dans la formulation du problème d'optimisation. Ces exigences peuvent être par exemple des critères perceptifs. Ces exigences sont alors exploitées dans une dernière étape qui consiste à prendre une décision en termes de choix de conception. Cette décision se fait sur les solutions optimales générées par l'algorithme d'optimisation et via l'interaction directe du concepteur avec ces solutions.

Une représentation schématique différente de cette méthode d'optimisation d'agencement est décrite également dans (BPBR11a). Dans cette étude, la démarche d'optimisation est testée sur un des cas d'application qui sera présenté dans le chapitre 5.

2 Description et formulation des problèmes d'optimisation d'agencement

2.1 Introduction

Ce chapitre décrit les deux premières étapes de la méthode d'optimisation proposée dans ce manuscrit. L'objectif global de ces deux étapes consiste à transformer le besoin du concepteur en un problème d'optimisation d'agencement d'espace. Pour cela, la démarche d'optimisation propose une première étape de description du problème puis une seconde étape de formulation du problème d'agencement.

La première partie de ce chapitre est donc consacrée à la méthode qui permet de décrire les problèmes d'agencement, afin de construire un premier modèle géométrique de l'agencement étudié. Ensuite, la seconde partie du chapitre détaillera la manière de traduire les exigences du concepteur en variables, contraintes et objectifs d'optimisation. Dans cette partie, une méthode innovante pour exprimer quantitativement la contrainte d'accessibilité aux composants d'un agencement est notam-

ment proposée. Enfin, la dernière partie de ce chapitre définit un nouvel indicateur permettant au concepteur de savoir à priori si son problème d'optimisation d'agencement peut être ou non résolu.

2.2 Description d'un problème d'agencement

Cette partie propose une méthode permettant de décrire de manière générique un problème d'agencement d'espace, afin de créer un modèle géométrique de cet agencement. Ce modèle géométrique est notamment basé sur une nouvelle classification des composants de l'agencement.

2.2.1 Composants « matériels » et « virtuels »

Un problème d'agencement consiste de manière générale à placer un certain nombre de composants dans un contenant. L'idée principale est ici de décomposer les composants présents dans le contenant en deux catégories (BBPR10d) :

- les composants « **matériels** » : ont une masse et ne peuvent pas se chevaucher entre eux,
- les composants « **virtuels** » : n'ont pas de masse et peuvent ou non chevaucher d'autres composants, selon les exigences définies par le concepteur.

Il est important de faire cette classification des composants car cette distinction permet de décrire de manière générique tous les problèmes d'agencement, quelles que soient les exigences exprimées par le concepteur. En effet, considérons un problème d'agencement très simple en deux dimensions, illustré sur la figure 2.1. Ce problème porte sur l'optimisation d'agencement d'un local (modélisé par le contenant noir sur la figure 2.1), dans lequel se trouvent trois armoires. Les armoires 2 et 3 sont de taille identique.

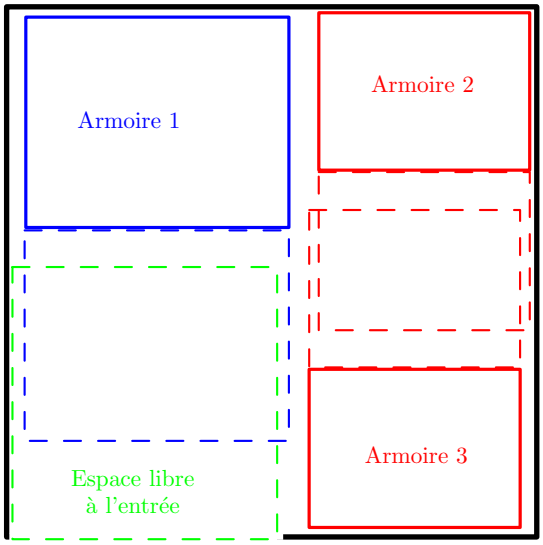


FIGURE 2.1 – *Problème d'agencement simple avec des composants matériels et virtuels*

Supposons que le concepteur souhaite pouvoir manipuler ces trois armoires (par exemple ouvrir la porte pour y insérer du matériel) sans avoir à déplacer les autres équipements. De même, la manipulation de toutes les armoires n'est pas nécessairement réalisée simultanément. Dans ce cas, la décomposition des composants de cet agencement en composants matériels et virtuels permet de prendre en compte cette exigence particulière exprimée par le concepteur. En effet, les armoires sont modélisées par des composants matériels car elles ne peuvent pas se chevaucher entre elles. Les espaces libres, placés à proximité des armoires, et utiles à la manipulation de celles-ci (ouverture de la porte), sont modélisés par des composants virtuels. Ces composants virtuels peuvent se chevaucher entre eux, du fait de la manipulation non simultanée des armoires. Ces composants virtuels seront dans la suite du document appelés « espaces d'accessibilité ». Ils sont représentés sur la figure 2.1 par des rectangles bleus et rouges en trait pointillé.

Aussi, il est important de laisser un espace libre à proximité de l'entrée du contenant pour faciliter l'accès au local. Cet espace libre est lui aussi modélisé par un composant virtuel (rectangle vert en trait pointillé sur la figure 2.1). Les espaces d'accessibilité des armoires peuvent chevaucher cet espace libre.

Cette décomposition des composants se retrouve dans la plupart des problèmes d'agencement, où tous les composants n'ont pas les mêmes propriétés et ne sont pas soumis aux mêmes exigences fixées par le concepteur. On peut notamment citer les problèmes de placement de machines dans une usine (DPHG07). Ces composants virtuels permettent de modéliser les espaces libres autour des machines et nécessaires à l'utilisation de celles-ci par les opérateurs. Ils peuvent également modéliser les couloirs disposés dans l'usine et utilisés pour le flux de marchandises entre les équipements. Aussi, on peut citer les problèmes d'assemblage de composants d'un mécanisme. Les composants virtuels peuvent modéliser les espaces d'accessibilité placés autour de certains composants et utiles au bon fonctionnement de ces composants ou à leur maintenance.

2.2.2 Attributs de forme des composants

Il est possible également de détailler davantage la classification proposée au paragraphe précédent en décomposant les composants suivant leurs attributs de forme. Ces attributs sont divisés en trois catégories :

- **forme fixe** : la dimension ou la forme du composant ne change pas au cours du processus d'optimisation d'agencement. La plupart des problèmes d'agencement d'espace traités dans la littérature considèrent ce type de composants,
- **dimension(s) variable(s)** : une ou plusieurs dimensions de un ou plusieurs composants peuvent varier au cours du processus d'optimisation d'agencement. Dans l'exemple illustré sur la figure 2.1, on pourrait supposer que les dimensions de l'armoire 1 sont variables et que le concepteur souhaite maximiser la taille de cette armoire. Aussi, un cas d'application traité dans le chapitre 5 présente une problème d'optimisation d'agencement où les dimensions du contenant sont variables,
- **forme variable** : la forme de un ou plusieurs composants varie au cours du processus d'optimi-

sation d'agencement. Ce type d'attribut peut s'appliquer par exemple à une chaise pliante que l'on souhaite placer dans un local. Les dimensions de cette chaise ne varient pas mais sa forme peut changer (pliée ou dépliée). Ce composant peut donc faire l'objet d'un traitement particulier lors de l'évaluation des contraintes et des objectifs de conception. Le premier cas d'application, présenté dans le chapitre 5, traite le placement d'une table pliante qui peut se placer devant une armoire car lorsqu'elle est pliée, elle n'empêche pas l'accès à l'armoire.

Un même composant peut avoir à la fois les deux derniers attributs : dimension(s) variable(s) et forme variable. Par exemple, dans un problème d'aménagement d'un salon, le concepteur peut souhaiter optimiser le placement d'un canapé qui a deux positions (assis ou couchage) tout en maximisant sa longueur. Aussi, certains problèmes d'optimisation d'agencement nécessitent une adaptation des dimensions et de la forme de certains composants afin de respecter toutes les contraintes de conception. Une stratégie d'optimisation dédié à ces problèmes est proposée dans (DGF11).

2.2.3 Classification des composants

En résumé, il est possible d'établir une classification générique et complète des composants d'un problème d'agencement. Cette classification s'appuie sur la décomposition en composants matériels et virtuels et sur les attributs de forme énoncés dans les deux paragraphes précédents. Cette classification est illustrée sur la figure 2.2. Un exemple est donné à titre indicatif pour chaque catégorie de composants.

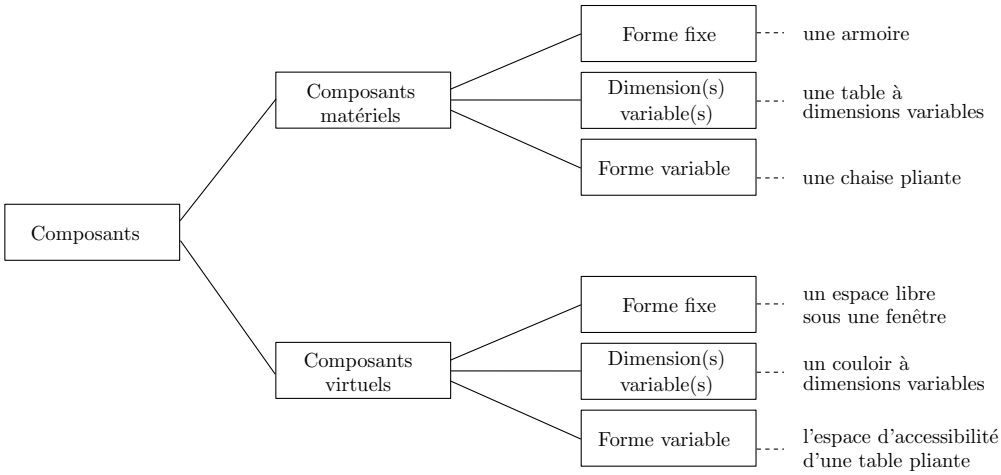


FIGURE 2.2 – *Classification des composants d'un problème d'agencement d'espace*

Comme expliqué dans la sous-section précédente, un composant peut avoir à la fois des dimensions variables et une forme variable. Il peut donc être situé dans deux catégories différentes de la classification illustrée sur la figure 2.2. Finalement, cette classification générique des composants permet de créer un modèle géométrique de l'agencement qui traduit toutes les spécificités et toutes les exigences exprimées par le concepteur.

2.3 Formulation d'un problème d'agencement

2.3.1 Formulation générale

Dans la démarche de conception proposée dans ce manuscrit, une formulation multiobjectif du problème d'optimisation d'agencement est privilégiée. En effet, comme il l'est expliqué dans le chapitre 1, il existe deux possibilités pour formuler un problème d'optimisation : la formulation mono-objectif ou la formulation multiobjectif.

Dans la plupart des applications industrielles, les critères d'optimisation sont multiples et bien souvent contradictoires. Dans la formulation mono-objectif, le concepteur doit à priori exprimer l'ensemble des critères de performances de l'agencement en une seule fonction objectif. Par exemple, il peut faire une somme pondérée des différents critères d'optimisation. Cependant, la création de cette fonction objectif unique est possible seulement si les différents critères d'optimisation ont la même dimension et que le concepteur a une très bonne connaissance de son problème, afin de fixer au mieux les coefficients de pondération utilisés dans la fonction objectif.

Pour ces raisons, la méthode proposée ici privilégie le formulation multiobjectif du problème d'optimisation d'agencement. Plusieurs critères d'optimisation, parfois contradictoires, sont à améliorer simultanément. Le résultat issu de cette formulation n'est pas une solution optimale unique mais un ensemble de solutions Pareto-optimales. Il convient donc ensuite de fournir au concepteur un ensemble de méthodes et d'outils pour faire un choix de conception parmi ces solutions.

Comme il l'est expliqué dans le chapitre 1, un problème d'optimisation d'agencement multiobjectif peut se formuler de la façon suivante :

$$\text{Problème P} : \begin{cases} \text{Trouver les valeurs optimales des variables de conception } \mathbf{x}^* \\ \text{avec } \mathbf{x} = (x_1, x_2, ..., x_n) \in \mathbb{R}^n \\ \mathbf{x}^* \in \operatorname{argmin} \mathbf{F}(\mathbf{x}, \mathbf{p}) = \operatorname{argmin} \left(f_1(\mathbf{x}, \mathbf{p}), f_2(\mathbf{x}, \mathbf{p}), ...f_m(\mathbf{x}, \mathbf{p})\right) \\ \text{sous contraintes :} \\ \mathbf{h}(\mathbf{x}^*, \mathbf{p}) = 0 \\ \mathbf{g}(\mathbf{x}^*, \mathbf{p}) \leqslant 0 \end{cases} \tag{2.1}$$

où n représente le nombre de variables d'optimisation et m le nombre d'objectifs. Le vecteur $\mathbf{p}$ représente l'ensemble des paramètres du problème d'optimisation. Le vecteur de fonctions $\mathbf{F}$ regroupe les critères d'optimisation et les vecteurs de fonctions $\mathbf{h}$ et $\mathbf{g}$ définissent respectivement les contraintes d'égalité et d'inégalité du problème.

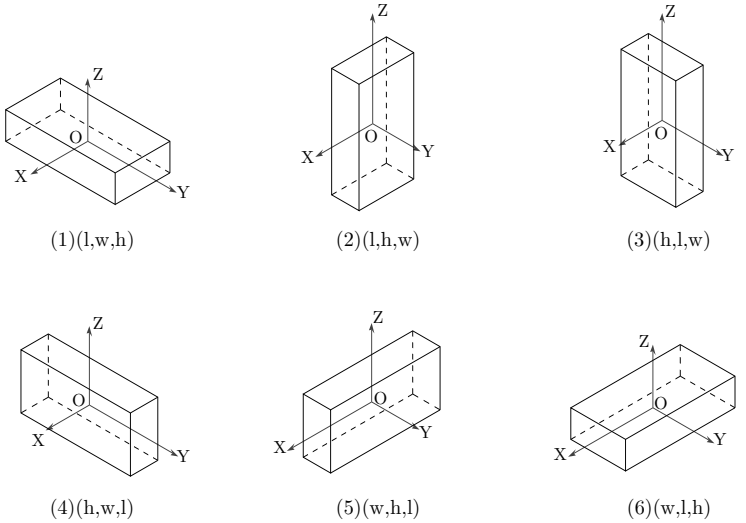
Étant donnée l'équation 2.1, trois éléments sont indispensables à la formulation d'un problème d'optimisation d'agencement : les variables d'optimisation, les contraintes de conception et les objectifs.

2.3.2 Variables d'optimisation

Les variables d'optimisation sont les paramètres qui vont être modifiés durant le phase d'optimisation. Ces paramètres influent sur les performances globales de l'agencement et leur valeur sera fixée à la fin du processus d'optimisation. Dans la plupart des problèmes d'agencement, ces variables de conception sont les variables qui localisent les composants à l'intérieur du contenant.

Dans ce manuscrit de thèse, les problèmes d'agencement qui sont traités ont tous des composants de forme rectangulaire pour les cas en deux dimensions ou parallélépipédique pour les cas en trois dimensions. Les variables d'optimisation peuvent donc être définies par :
- des variables continues (X,Y,Z) qui caractérisent la position du centre géométrique de chaque composant,
- une variable discrète α qui modélise l'orientation du composant,
- une variable discrète λ qui modélise le sens du composant.

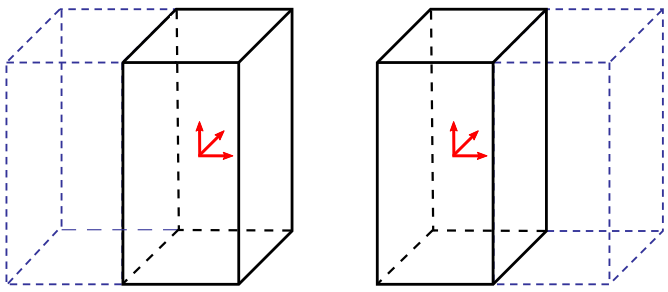
Les variables de positionnement du centre géométrique du composant peuvent être considérées comme des variables discrètes, selon les exigences définies par le concepteur (par exemple lorsqu'un composant est fixé sur un des murs d'un local). Aussi, il est possible de considérer différents modes de représentation de l'orientation (α) du composant. Dans la suite du document, nous opterons pour la modélisation illustrée sur la figure 2.3. La variable α peut prendre six valeurs (de 1 à 6) correspondant aux six orientations possibles d'un parallélépipède dans l'espace. Pour les problèmes d'agencement en deux dimensions dans le plan $(O, \vec{X}, \vec{Y})$, les valeurs possibles pour α sont restreintes à 1 ou 6.



	1	2	3	4	5	6
Dimensions	(l,w,h)	(l,h,w)	(h,l,w)	(h,w,l)	(w,h,l)	(w,l,h)
Permutations	(1,2,3)	(1,3,2)	(3,1,2)	(3,2,1)	(2,3,1)	(2,1,3)

FIGURE 2.3 – *Orientations possibles d'un parallélépipède de dimensions (l,w,h) dans le repère $(0,\vec{X},\vec{Y},\vec{Z})$. Image empruntée à Tiwari et al. (TFF08)*

La variable discrète λ peut prendre deux valeurs 1 ou 2 et est utilisée uniquement pour les composants qui ont un espace d'accessibilité, comme il l'est expliqué à la section précédente. En effet, pour une orientation donnée du parallélépipède, il existe deux sens possibles, correspondant aux deux placements possibles de l'espace d'accessibilité. Le composant matériel et son espace d'accessibilité ont une face entière commune et le placement de l'espace d'accessibilité est lié au placement du composant matériel via cette variable discrète λ. La figure 2.4 illustre les deux sens possibles pour un composant qui a une orientation donnée.



(a) $\lambda = 1$ (b) $\lambda = 2$

FIGURE 2.4 – *Placement de l'espace d'accessibilité par la variable discrète* λ

Cette modélisation des variables d'optimisation, adaptée aux problèmes d'agencement avec des composants de forme parallélépipédique, est suffisamment générique pour prendre en compte les spécificités de chaque problème. Il est possible de ne pas utiliser la variable λ qui définit le sens d'un composant à condition d'ajouter deux fois plus de valeurs (c'est-à-dire 12 valeurs) pour la variable d'orientation α. En résumé, il faudrait décomposer chaque cas présenté sur la figure 2.3 en deux sous-cas, correspondant aux deux sens possibles du composant.

Aussi, le nombre de variables d'optimisation caractérise le nombre de degrés de liberté associé au problème d'agencement. Certaines variables peuvent être constantes, ce qui réduit le nombre de variables d'optimisation et donc facilite la recherche de solutions. Enfin, dans certaines applications, quand un ou plusieurs composants ont une ou plusieurs dimensions variables, le concepteur utilise des variables d'optimisation supplémentaires pour prendre en compte ces dimensions.

2.3.3 Contraintes et objectifs de conception

Il est tout d'abord important de mentionner ici que, dans cette partie, les termes « contrainte » et « objectif » peuvent être échangés. En effet, le concepteur, bien souvent non spécialisé dans le domaine des méthodes d'optimisation, exprime uniquement des exigences qui sont formulées en contraintes ou en objectifs, selon le degré de priorité de ces exigences. Dans la suite de ce paragraphe, nous utiliserons à titre indicatif le terme « contrainte ».

Deux catégories de contraintes sont considérées : les **contraintes géométriques** et les **contraintes fonctionnelles**. Les contraintes géométriques garantissent essentiellement le non-chevauchement

entre composants. La détection de la collision entre composants est calculée à chaque itération de l'algorithme d'optimisation. Il est donc important de choisir une bonne représentation des composants et une méthode efficace de détection afin de réduire les temps de calcul. On peut trouver dans (LG98) un résumé des principales méthodes utilisées dans la détection de collisions. Dans ce manuscrit, pour les problèmes d'agencement en trois dimensions, étant donnée la forme parallélépipédique des composants, les contraintes de non-chevauchement peuvent être évaluées via le calcul du volume d'intersection entre les composants. Ce volume d'intersection, évalué entre les composants i et j est calculé par la formule suivante :

$$\begin{aligned} V_{ij} &= \max[0, \min(x_i + \frac{l_i}{2}, x_j + \frac{l_j}{2}) - \max(x_i - \frac{l_i}{2}, x_j - \frac{l_j}{2})] \\ &\quad \times \max[0, \min(y_i + \frac{L_i}{2}, y_j + \frac{L_j}{2}) - \max(y_i - \frac{L_i}{2}, y_j - \frac{L_j}{2})] \\ &\quad \times \max[0, \min(z_i + \frac{H_i}{2}, z_j + \frac{H_j}{2}) - \max(z_i - \frac{H_i}{2}, z_j - \frac{H_j}{2})] \end{aligned} \tag{2.2}$$

où les valeurs (x_i, y_i, z_i) représentent les coordonnées du centre géométrique du composant i. l_i, L_i et H_i définissent respectivement les dimensions du composant i suivant les axes $(O, \vec{X})$, $(O, \vec{Y})$ et $(O, \vec{Z})$. Pour les problèmes en deux dimensions, où les composants sont de forme rectangulaire, ce volume d'intersection est réduit à l'aire d'intersection entre composants.

Dans certains problèmes d'agencement, les contraintes et les objectifs sont seulement géométriques. Ces problèmes peuvent être assimilés aux problèmes de découpe et de conditionnement (Dyc90; WHS07). D'un autre côté, on peut trouver des contraintes fonctionnelles qui garantissent le bon fonctionnement de l'agencement. Ces contraintes sont multiples : équilibre des masses, distance maximale entre composants, alignement des composants, accessibilité aux composants.... L'idée est donc de proposer un ensemble de fonctions mathématiques génériques qui permettent de traduire ces contraintes.

Aussi, pour certains problèmes, certaines contraintes sont difficilement traduisibles en fonctions mathématiques explicites. Ces contraintes peuvent par exemple exprimer les connaissances métier du concepteur, son expertise ou encore ses perceptions. La partie suivante propose une méthode pour traduire une exigence particulière : l'accessibilité aux composants depuis l'entrée d'un contenant. On retrouve cette exigence dans de nombreux problèmes d'agencement, tels l'agencement d'équipements dans un local (cas d'application traité dans le chapitre 5), le placement de machines dans une usine ou encore l'assemblage de pièces d'un mécanisme. Pour le problème d'agencement de pièces dans un assemblage mécanique, cette exigence d'accessibilité traduit par exemple le fait qu'un composant doit rester accessible pour maintenance éventuelle.

2.3.4 Cas particulier de l'accessibilité aux composants

L'accessibilité aux composants d'un agencement est une exigence que l'on retrouve dans de nombreux problèmes d'optimisation d'agencement (BBPR10a). Cette exigence doit garantir le fait qu'un composant doit être accessible depuis l'entrée du contenant. Deux manières de prendre en compte cette contrainte sont possibles :

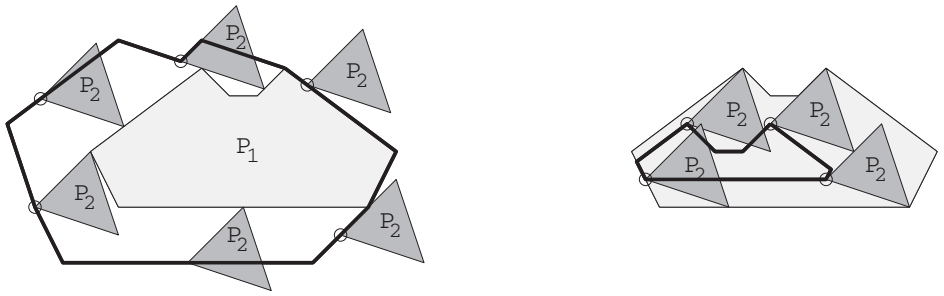
- modifier la description du problème, c'est-à-dire le modèle géométrique, afin d'assurer l'accessibilité aux composants. Le concepteur peut par exemple ajouter un ou plusieurs composants virtuels dans l'agencement (espace libre, couloir...) permettant d'accéder aux composants,
- intégrer cette contrainte d'accessibilité à la formulation du problème d'optimisation d'agencement, c'est-à-dire traduire cette exigence en contrainte ou en objectif d'optimisation.

La première possibilité est la plus facile à mettre en œuvre. Cependant, cette approche ne permet pas la recherche de solutions innovantes. En effet, les solutions générées par l'algorithme d'optimisation vont naturellement se rapprocher du modèle géométrique proposé par le concepteur. Par exemple, si le concepteur décide d'ajouter un couloir central dans son local afin d'avoir accès à tous les composants, toutes les solutions trouvées par l'algorithme auront un couloir central. Par conséquent, nous nous intéressons ici à la deuxième approche, qui favorise la recherche de solutions innovantes. Deux méthodes sont proposées afin de prendre en compte l'exigence d'accessibilité dans la formulation du problème d'optimisation d'agencement. Ces deux méthodes sont adaptées à la résolution des problèmes d'agencement en deux dimensions. Les deux stratégies sont expliquées sur un exemple d'agencement d'équipements dans un local. Cet exemple sera développé davantage dans le chapitre 5, relatif aux applications industrielles.

2.3.4.1 Approche basée sur le polygone d'appartenance

Cette approche s'appuie sur le calcul du polygone d'appartenance (*Inner-Fit-Polygon* en anglais). Le polygone d'appartenance dérive du polygone de non-recouvrement (*No-Fit-Polygon* en anglais). Le polygone de non-recouvrement entre les polygones $P1$ et $P2$ détermine l'ensemble des positions d'un point de référence du polygone $P2$, tel que celui-ci touche le polygone $P1$ mais ne rentre jamais en intersection avec celui-ci. Pour construire ce polygone de non-recouvrement, nous supposons que le polygone $P2$ se déplace seulement en translation. La figure 2.5 illustre le polygone de non-recouvrement et le polygone d'appartenance (en trait épais noir sur la figure) entre deux polygones $P1$ et $P2$.

Afin d'utiliser le principe du polygone d'appartenance pour prendre en compte l'exigence d'accessibilité aux composants, considérons tout d'abord qu'en deux dimensions, le concepteur est modélisé par un cercle C de rayon R. La valeur du rayon R est fixée arbitrairement par le concepteur. Le centre de ce cercle C est représenté par un point A. D'autre part, le contenant est modélisé par un polygone rectangulaire P dans lequel tous les composants de l'agencement sont également des polygones rectangulaires, mais considérés comme des trous à l'intérieur du polygone P. Ce polygone



(a) polygone de non-recouvrement

(b) polygone d'appartenance

FIGURE 2.5 – *Polygone de non-recouvrement (a) et d'appartenance (b) de deux polygones P1 et P2*

P regroupe donc le contenant et tous les composants modélisés par des trous.

L'approche proposée ici suggère de calculer le polygone d'appartenance du cercle C et du polygone P, c'est-à-dire toutes les positions du point A, telles que le cercle C soit à l'intérieur du polygone P, sans chevaucher les composants. Ce polygone d'appartenance, défini par P_{IFP}, peut être composé de plusieurs polygones. Le polygone, compris dans P_{IFP}, qui est le plus proche de l'entrée du contenant est sélectionné. Ce polygone est identifié par le terme P^*_{IFP}. Ensuite, un point d'accès Pa_i est défini pour chaque composant pour lequel on souhaite vérifier l'accessibilité depuis l'entrée du contenant. La distance D_i entre Pa_i et P^*_{IFP} est mesurée. Si cette distance est supérieure au rayon R du cercle C, alors le composant i n'est pas accessible depuis l'entrée du contenant.

La figure 2.6, illustre cette approche pour deux exemples d'agencement. Le polygone P_{IFP} est tracé en trait rouge sur la figure et la distance entre chaque point d'accès Pa_i aux composants et le polygone P^*_{IFP} est modélisé par un trait fin noir. Sur cette figure, la face d'accès à un composant est marquée d'un trait vert. Le point d'accès Pa_i est fixé au centre de la face d'accès au composant i. Dans cet exemple, le concepteur souhaite assurer l'accessibilité aux composants numérotés de 1 à 6 sur la figure 2.6. On remarque, sur une des deux illustrations de la figure 2.6, que le polygone d'appartenance est composé de deux polygones et que le polygone le plus proche de l'entrée du contenant est sélectionné dans l'évaluation de l'accessibilité aux composants.

Afin de prendre en compte cette exigence d'accessibilité dans la formulation du problème d'optimisation d'agencement, il convient d'exprimer cette exigence par une donnée quantitative qui pourra être considérée comme une contrainte ou un objectif du problème d'optimisation. Cette donnée est exprimée par la formule suivante :

$$Acc = \max\left(D_i - R\right), \textit{ pour } i \in \{1, ..., n\} \tag{2.3}$$

n représentant le nombre de composants pris en compte dans ce calcul d'accessibilité.

Finalement, si l'accessibilité aux composants depuis l'entrée du contenant est une contrainte du

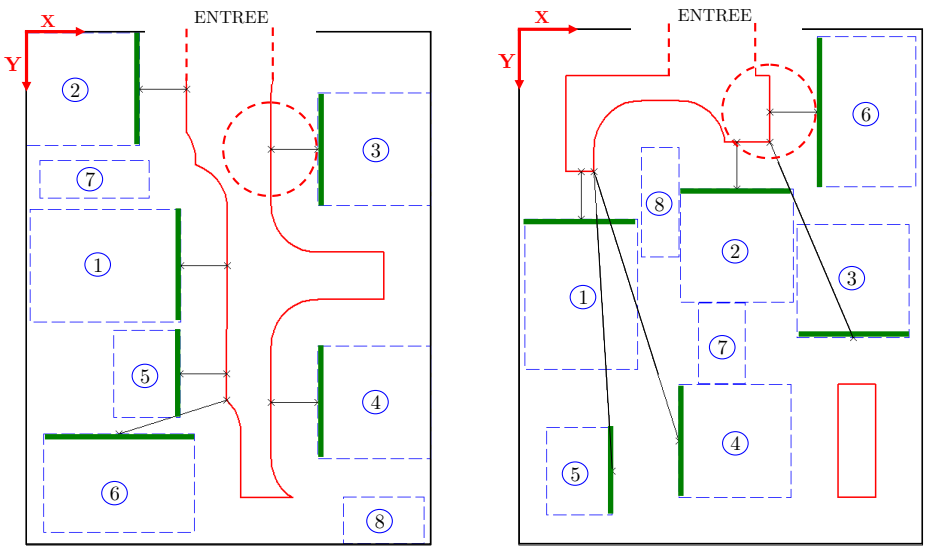


FIGURE 2.6 – *Accessibilité aux composants : approche basée sur le polygone d'appartenance*

problème d'optimisation, la condition suivante est appliquée : $Acc \leqslant 0$. Si cette même exigence est formulée par un objectif d'optimisation, cet objectif est formulé par : $F = \min(Acc)$.

2.3.4.2 Approche basée sur l'optimisation de trajectoire

Cette seconde approche est basée sur un processus d'optimisation de trajectoire. La trajectoire est définie entre deux points : le point E qui est localisé à l'entrée du contenant et le point d'accès Pa_i modélisé pour chaque composant. Nous considérons que cette trajectoire est composée de n_p portions de trajectoires, de forme rectangulaire et liées entre elles. Cette trajectoire modélise par exemple le trajet emprunté par le concepteur pour se rendre de l'entrée du contenant au point d'accès du composant.

Ce problème d'optimisation de trajectoire est un problème d'optimisation mono-objectif sous contraintes, qui peut se formuler ainsi :

$$\text{Problème P} : \begin{cases} trouver\ \mathbf{x}^* = (l_{p1}, l_{p2}, ..., l_{p(n_p-2)}, \theta_{p1}, \theta_{p2}, ..., \theta_{p(n_p-2)}) \\ \mathbf{x}^* = \operatorname{argmin} F(\mathbf{x}) \\ \text{sous contraintes :} \\ \mathbf{g}(\mathbf{x}^*) \leqslant 0 \end{cases} \tag{2.4}$$

Les variables d'optimisation correspondent aux longueurs (l_p) et aux orientations (θ_p) des portions de la trajectoire. Seulement les $(n_p - 2)$ premières portions de trajectoires sont pilotées par l'optimisation. La variable représentant la longueur d'une portion est continue alors que la variable représentant l'orientation est discrète et peut prendre les valeurs 0°, 90°, 180° ou 270°, correspon-

dant aux quatre orientations possibles de la portion de trajectoire en deux dimensions. Finalement, le nombre de variables d'optimisation est égal à $2 \times (n_p - 2)$. La figure 2.7 illustre les quatre orientations possibles d'une portion de trajectoire (portion en noir numérotée 1) et montre comment la portion suivante (portion en bleu numérotée 2) est attachée à la portion précédente.

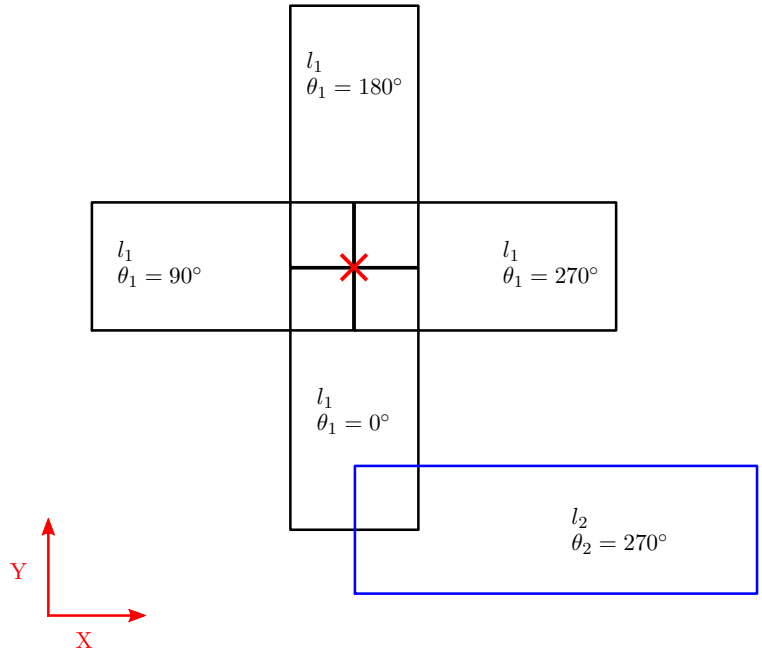


FIGURE 2.7 – *Définition du placement en 2D des portions de la trajectoire : longueur et orientation*

Le vecteur des contraintes d'inégalité $\mathbf{g}$ caractérisent les contraintes de non-chevauchement des différentes portions de la trajectoire avec les composants de l'agencement et l'extérieur du contenant. Ces contraintes de non-chevauchement peuvent être évaluées via l'aire d'intersection entre deux éléments rectangulaires, formulée dans l'équation 2.2. L'objectif d'optimisation consiste à minimiser la longueur totale de la trajectoire, c'est-à-dire la somme des longueurs des portions qui composent la trajectoire.

La figure 2.8 illustre cette approche pour un exemple d'agencement d'espace, où le concepteur souhaite évaluer l'accessibilité à deux composants différents. Sur une des deux figures, on teste l'accessibilité au composant n°4 et sur l'autre figure, l'accessibilité au composant n°6. Le paramètre n_p, représentant le nombre de portions composant la trajectoire, est fixé à 5. La trajectoire est modélisée par les rectangles en trait rouge.

Afin de prendre en compte cette exigence d'accessibilité dans la formulation du problème d'optimisation d'agencement, il convient, comme dans la première approche, de caractériser l'accessibilité par une donnée quantitative. Pour chaque composant i, la somme totale A_i des aires d'intersection entre les composants et les portions de la trajectoire est défini par la formule suivante :

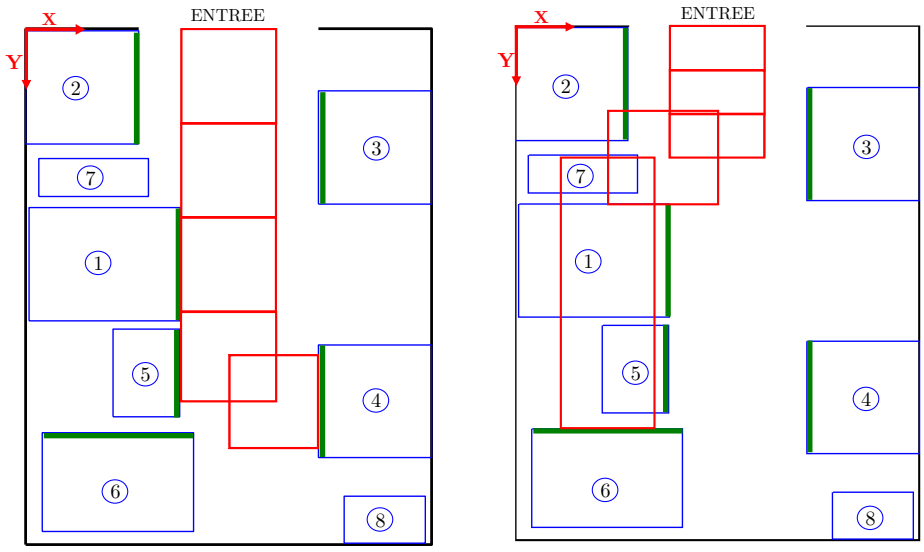


FIGURE 2.8 – *Accessibilité aux composants : approche basée sur l'optimisation de trajectoire*

$$A_i = \sum_{k=1}^{n} (\sum_{j=1}^{n_p} A_{kj}) \tag{2.5}$$

où A_{kj} évalue l'aire d'intersection entre la portion de trajectoire k et le composant j. En résumé, si A_i est égale à 0, alors le composant i est accessible depuis l'entrée du contenant. Par conséquent, la donnée quantitative qui évalue l'exigence d'accessibilité à tous les composants est définie par :

$$Acc = \sum_{i=1}^{n} A_i \tag{2.6}$$

Finalement, si l'accessibilité aux composants depuis l'entrée du contenant est une contrainte du problème d'optimisation, la condition suivante est appliquée : $Acc \leqslant 0$. Si cette même exigence est formulée par un objectif d'optimisation, cet objectif est formulé par : $F = \min(Acc)$.

2.3.4.3 Comparaison des deux approches

Les deux approches ont été testées sur dix problèmes d'agencement différents, similaires à ceux illustrés sur les figures 2.6 et 2.8. Pour chaque problème, le temps de calcul de chaque méthode est évalué et la valeur de l'accessibilité aux composants est mesurée.

Pour l'approche basée sur le polygone d'appartenance, le rayon R est fixé à 50 cm. Pour l'approche basée sur l'optimisation de trajectoire, la trajectoire est composée de 5 portions. Chaque largeur de portion est égale à 50 cm. L'algorithme d'optimisation utilisé pour réaliser l'optimisation de trajectoire est basé sur une méthode de programmation quadratique séquentielle (*Sequential Quadratic Programming* - SQP, en anglais) (NW99). Cet algorithme n'est pas spécifiquement l'algorithme

d'optimisation le plus adapté pour ce problème d'optimisation, compte tenu des variables discrètes d'orientation des portions de la trajectoire. Cependant, les résultats obtenus sur l'exemple testé sont très satisfaisants. Aussi, il existe d'autres méthodes d'optimisation adaptées à ce problème d'optimisation de trajectoire (programmation par contrainte, algorithme génétique...) L'algorithme SQP est initialisé par le vecteur $[L, L, L, 0°, 0°, 0°]$, avec L fixé arbitrairement à 50 cm. Le nombre maximal d'itérations autorisé pour l'algorithme d'optimisation est fixé à 100.

Il n'existe pas de différence entre les deux méthodes sur la valeur de l'accessibilité aux composants. Chaque méthode détecte bien quel composant est accessible depuis l'entrée du contenant. La différence se fait sur le temps de calcul. Le temps de calcul moyen mesuré pour l'approche basée sur le polygone d'appartenance est égale à 0,41s par problème d'agencement traité. Ce temps de calcul est inférieur à celui mesuré pour l'approche basé sur l'optimisation de trajectoire qui est égal à 0,99s. Cette différence peut avoir une grande importance sur le temps de calcul du processus global d'optimisation d'agencement, car cette contrainte d'accessibilité est calculée à chaque évaluation d'une nouvelle solution. Cependant, le temps de calcul de la méthode basée sur l'optimisation de trajectoire peut être réduit en réduisant la complexité du problème (en réduisant par exemple le nombre de portions de la trajectoire, c'est-à-dire le nombre de variables d'optimisation).

Aussi, les différentes simulations effectuées montrent que le temps de calcul obtenu pour l'approche basée sur le polygone d'appartenance fluctue très peu d'une simulation à l'autre. Pour l'autre approche, le temps de calcul varie beaucoup. Ceci est dû au fait que cette approche utilise un processus d'optimisation et que le temps de calcul de l'algorithme d'optimisation dépend fortement des conditions initiales, telles la trajectoire initiale et la position du composant dont l'accessibilité est testée. Par conséquent, dans l'application présentée dans le chapitre 5, nous privilégierons la première approche basée sur le polygone d'appartenance.

Enfin, pour certains problèmes d'agencement, comme dans un assemblage mécanique, le concepteur ne peut pas entrer dans l'agencement. Seulement un outil, un robot ou la main du concepteur, doivent avoir accès à certains composants. Il est donc nécessaire d'adapter les deux approches proposées dans cette sous-section aux contraintes que posent ces problèmes d'accessibilité particuliers.

2.4 Indicateur de faisabilité d'un problème d'agencement

Cette partie propose une méthode permettant de définir un indice de faisabilité d'un problème d'agencement, c'est-à-dire un indice permettant au concepteur de savoir à priori si le problème d'agencement peut être ou non résolu. Cet indice de faisabilité est proche de l'indice de compacité d'un problème d'agencement. Pour rappel, la compacité d'un problème d'agencement en trois dimensions est définie par :

$$C = \frac{Espace\,occupé}{Espace\,total} = \frac{\sum_{i=1}^{n} V_i}{V_{contenant}} \tag{2.7}$$

où n définit le nombre de composants à placer dans le contenant et V_i représente le volume occupé par le composant i. Pour les problèmes d'agencement en deux dimensions, le volume V est remplacé par l'aire A du composant.

Cette définition de la compacité ne prend pas en compte ni la classification des composants présentée dans le section 2.2, ni les relations géométriques et fonctionnelles qui peuvent exister entre les composants. En effet, considérons l'exemple illustré sur la figure 2.1. Les dimensions du contenant et des différents composants de l'agencement sont définies dans le tableau 2.1. On remarque sur ce tableau que les espaces d'accessibilité ont les mêmes dimensions que les composants associés.

Numéro	Contenant et composants	Dim /X	Dim /Y
0	Contenant	500	500
1	Armoire 1	250	200
2	Armoire 2	200	150
3	Armoire 3	200	150
4	Espace d'accessibilité de l'armoire 1	250	200
5	Espace d'accessibilité de l'armoire 2	200	150
6	Espace d'accessibilité de l'armoire 3	200	150
7	Espace libre à l'entrée	250	250

TABLE 2.1 – *Dimensions des composants du problème d'agencement illustré sur la figure 2.1*

En appliquant la formule de la compacité, définie par l'équation 2.7, et en considérant qu'on calcule des aires et non des volumes, on obtient une compacité égale à 113 %. Ceci signifie que l'optimisation d'agencement ne peut, à priori, pas être résolue car l'espace disponible à l'intérieur du contenant est inférieur à l'espace total occupé par tous les composants. Cependant, la figure 2.1 montre bien qu'il existe au moins une solution qui respecte toutes les contraintes définies par le concepteur. Par conséquent, cette définition de la compacité ne permet pas d'établir un indicateur fiable de faisabilité pour les problèmes d'agencement qui associent des composants matériels et des composants virtuels.

Il convient donc de modifier cet indice de compacité afin que celui-ci prenne en compte les différentes contraintes de placement formulées par le concepteur, notamment entre les composants virtuels et les autres composants. Ce nouvel indice, que nous appellerons « indice de faisabilité » doit permettre de montrer, de manière fiable, la faisabilité ou non d'un problème d'optimisation d'agencement. L'objectif est en effet de trouver un indice qui, s'il est supérieur à 100 %, indique alors que le problème d'agencement d'espace ne peut pas être résolu.

La méthode permettant de construire cet indice de faisabilité est expliquée ici. Construisons, dans un premier temps, la matrice M_{inter} correspondant à la matrice d'intersections (encore appelée matrice de chevauchements) des composants entre eux. Chaque composante de la matrice symétrique M_{inter} est définie par :

$$\begin{cases} \forall\, i \in \{1, ..., n\}\,, \forall\, j \in \{1, ..., n\}\,, \\ M_{inter}(i,j) = \begin{vmatrix} 1 \text{ si intersection interdite} \\ 0 \text{ si intersection possible} \end{vmatrix} \end{cases} \tag{2.8}$$

L'équation 2.8 signifie donc que si le composant i et le composant j ne peuvent pas se chevaucher, le terme $M_{inter}(i,j)$ est égal à 1. Sinon, il est égal à 0. Aussi, nous considérons que la diagonale d'une matrice d'intersections est toujours un vecteur nul, indiquant qu'un composant se chevauche avec lui-même. Prenons l'exemple présenté sur la figure 2.1. On peut définir la matrice d'intersections de ce problème d'agencement. Celle-ci est présentée dans l'équation 2.9.

$$M_{inter} = \begin{vmatrix} 0 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 0 \end{vmatrix} \tag{2.9}$$

Dans la matrice d'intersections représentée dans l'équation 2.9, les composants sont classés dans le même ordre que celui indiqué dans le tableau 2.1. La 3$^{\text{ème}}$ ligne de cette matrice indique par exemple que le composant n°3 (armoire 3) ne peut chevaucher aucun autre composant. En analysant cette matrice d'intersections, il est possible de séparer les composants en deux groupes :

- groupe 1 : les composants qui ne peuvent pas chevaucher d'autres composants, c'est-à-dire les lignes (ou colonnes) avec un nombre de 0 égal à 1.
- groupe 2 : les composants qui peuvent se chevaucher entre eux, c'est-à-dire les lignes (ou colonnes) avec un nombre de 0 strictement supérieur à 1.

Dans l'exemple présenté dans ce chapitre, la matrice d'intersections permet de rassembler dans le groupe 1 les composants matériels (les trois armoires) et dans le groupe 2 tous les composants virtuels. Ensuite, l'indice de faisabilité du problème d'agencement est calculé via la formule suivante :

$$Indice_{faisabilité} = \frac{Espace\,occupé}{Espace\,total} = \frac{Espace\,occupé\,(groupe\,1) + Espace\,occupé\,(groupe\,2)}{Espace\,total} \tag{2.10}$$

L'espace occupé par le groupe 1 (composants qui ne peuvent pas chevaucher d'autres composants) est égal à la somme des volumes (ou aires en deux dimensions) des composants. Le calcul de l'espace occupé par les composants du groupe 2 (composants qui peuvent chevaucher d'autres composants) est plus complexe. Il s'agit de trouver l'espace minimal occupé par ces composants. On peut dans un premier temps supposer que tous les composants du groupe 2 se chevauchent entre eux et l'espace

minimal occupé par ces composants est égal au volume du composant le plus grand. C'est-à-dire qu'il est possible de placer les composants du groupe 2 dans le composant du groupe 2 qui a la plus grande taille. L'indice de faisabilité est alors calculé par la formule suivante :

$$Indice_{faisabilité} = \frac{Espace\,occupé}{Espace\,total} = \frac{\sum\limits_{i \in groupe1} V_i + \max\limits_{j \in groupe2} V_j}{V_{contenant}} \tag{2.11}$$

Cet indice présente quelques défauts car il fait des hypothèses arbitraires sur le chevauchement entre les composants du groupe 2 et ne prend donc pas en compte toutes les contraintes de placement formulées par le concepteur. L'autre manière d'évaluer l'espace occupé par le groupe 2 est de trouver une partition de l'ensemble des composants du groupe 2 qui minimise l'espace occupé tout en respectant les contraintes de chevauchement définies par la matrice d'intersections. Nous définissons la notion de partition comme étant un ensemble de sous-ensembles de composants, disjoints deux à deux et dont l'ensemble des composants forment le groupe 2. La figure 2.9 illustre un graphe représentant un exemple de partition avec deux sous-ensembles de composants. Chaque point de la figure représente un composant différent, appartenant au groupe 2. Le groupe 2 est, dans l'exemple, composé de 12 composants. Chaque arbre représente un sous-ensemble de composants (sous-ensemble 1 en rouge et sous-ensemble 2 en bleu). Les composants du niveau k sont inclus dans le composant qui leur est lié et situé au niveau $k-1$.

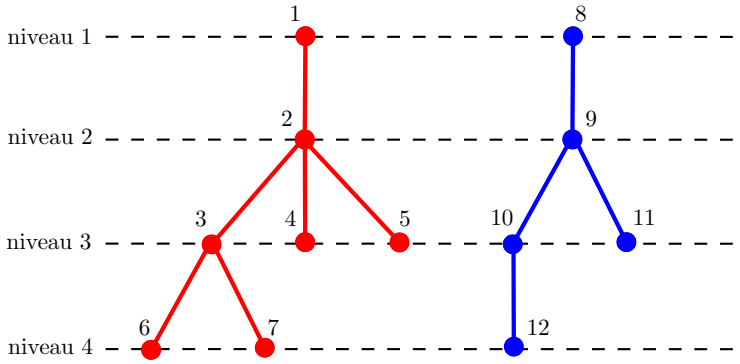


FIGURE 2.9 – *Exemple d'une partition de composants à 2 sous-ensembles*

En résumé, l'espace occupé par les composants du groupe 2 est la solution optimale du problème d'optimisation mono-objectif sous-contraintes formulé de la manière suivante :

$$\text{Problème P} : \begin{cases} \text{Trouver les valeurs optimales des variables de conception } \mathbf{x}^* \\ \text{avec } \mathbf{x} = (x_1, x_2, ..., x_{n_2}) \in \mathbb{Z}^{n_2} \\ \mathbf{x}^* = \operatorname{argmin} F(\mathbf{x}, \mathbf{p}) \\ \text{sous contraintes :} \\ \mathbf{g}(\mathbf{x}^*, \mathbf{p}) \leqslant 0 \end{cases} \tag{2.12}$$

où la variable n_2 définit le nombre de composants présents dans le groupe 2. Chaque variable x_i définit le sous-ensemble de composants auquel appartient le composant i. Il existe n_2 valeurs possibles pour chaque variable x_i, car avec n_2 composants, on peut former au maximum une partition de n_2 éléments où un élément est alors un sous-ensemble composé d'un seul composant. Par exemple, pour l'exemple de partition illustré sur la figure 2.9, le vecteur $\mathbf{x}$ est défini par $\mathbf{x} = (1\,1\,1\,1\,1\,1\,1\,2\,2\,2\,2\,2)$. Le vecteur $\mathbf{p}$ définit ici les autres paramètres du problème : le volume de chaque composant et la matrice d'intersections.

La fonction objectif F calcule l'espace occupé par la partition, c'est-à-dire la somme des espaces occupés par les sous-ensemble de composants. L'espace occupé par un sous-ensemble de composants est égal au volume du plus grand composant appartenant au sous-ensemble (valeur maximale des volumes des composants appartenant au sous-ensemble). La fonction objectif F s'exprime donc par :

$$\sum_{i=1}^{n_{SE}} (\max_{j \in SE_i} V_j) \tag{2.13}$$

où n_{SE} définit le nombre total de sous-ensembles de composants (taille de la partition). SE_i caractérise le sous-ensemble i de composants, et V_j le volume du composant j. En considérant le graphe présenté sur la figure 2.9, la fonction objectif F est égale à $V_1 + V_8$, c'est-à-dire la somme des volumes des composants situés au niveau 1. Aussi, sur ce graphe, la somme des volume des composants situés au niveau k est inférieure ou égale au volume du composant situé au niveau $(k-1)$ et qui leur est lié (par exemple, $V_3 + V_4 + V_5 \leqslant V_2$).

Dans le problème d'optimisation défini dans l'équation 2.12, il existe n_2 contraintes d'inégalité, qui évaluent la faisabilité des n_2 sous-ensembles possibles. Ces contraintes d'inégalité sont regroupées dans le vecteur des contraintes $\mathbf{g}$. On définit par g_k la contrainte d'inégalité qui évalue la faisabilité du sous-ensemble k de composants. Le calcul de la faisabilité d'un sous-ensemble de composants est une tâche difficile et nécessite l'utilisation d'algorithmes d'optimisation dédiés. Pour ne pas complexifier le calcul de l'indice de faisabilité, nous proposons une estimation rapide de cette contrainte de faisabilité g_k, en calculant la fonction g'_k. Cette fonction g'_k est définie de la manière suivante :

- $g'_k(\mathbf{x}, \mathbf{p}) = 0$ si l'une des deux conditions suivantes est vérifiée :
  - le sous-ensemble k est composé d'un seul composant,
  - tous les composants deux à deux peuvent se chevaucher (chevauchement possible indiqué dans la matrice d'intersections).
- $g'_k(\mathbf{x}, \mathbf{p}) = (V_i + V_j) - \max V_p$, pour $p, i, j \in SE_k$ et $p \notin \{i, j\}$ si :
  - deux composants i et j ne peuvent pas se chevaucher.

La fonction $g'_k(\mathbf{x}, \mathbf{p})$ est toujours inférieure ou égale à la fonction $g_k(\mathbf{x}, \mathbf{p})$, pour toute partition de composants. En effet, la contrainte d'inégalité g'_k est une contrainte plus faible car il existe des cas où $g'_k(\mathbf{x}, \mathbf{p}) = 0$ alors que le sous-ensemble k n'est pas réalisable. Par exemple, considérons, un sous-ensemble de 4 composants dont les volumes sont respectivement $V_1 = 2$, $V_2 = 1$, $V_3 = 1$ et

$V_4 = 1$. On suppose également que les composants 2, 3 et 4 peuvent chevaucher le composant 1 et que ces mêmes composants ne peuvent pas se chevaucher entre eux. La contrainte de faisabilité est vérifiée alors que le sous-ensemble n'est pas réalisable. Finalement, l'important est ici d'avoir une estimation de la contrainte de faisabilité du sous-ensemble de composants qui soit toujours inférieure à la contrainte de faisabilité réelle du sous-ensemble.

Afin de résoudre le problème d'optimisation indiqué dans l'équation 2.12, nous décidons d'utiliser un algorithme génétique (Omni-Optimizer de Deb *et al.* (DT08)). L'algorithme est initialisé par $16 \times n_2$ variables aléatoires. Le nombre maximal de générations de l'algorithme est fixé à 50. Les coefficients de mutation et de croisement pour ces variables discrètes sont respectivement réglés à 0,1 et 0,7. Pour prendre en compte le comportement stochastique de l'algorithme génétique, nous répétons le processus d'optimisation 10 fois. Après les 10 simulations réalisées, nous considérons alors la solution minimale (espace minimal occupé par le groupe 2) générée parmi les 10 simulations. Via cette estimation de l'espace minimal occupé par le groupe 2, nous pouvons en déduire alors l'indice de faisabilité du problème d'agencement, en utilisant la formule indiquée dans l'équation 2.10.

L'indice de faisabilité est calculé pour l'exemple d'agencement simple proposé dans ce chapitre. L'espace occupé par le groupe 1 (composants qui ne peuvent pas chevaucher d'autres composants) est donc égal à la somme des aires des trois armoires (=110 000). Pour calculer l'espace occupé par le groupe 2 (composants qui peuvent chevaucher d'autres composants), c'est-à-dire les composants virtuels, nous procédons à la résolution du problème d'optimisation présenté précédemment. Les 10 simulations donnent le même résultat (=62500). Étant donné que tous les composants virtuels peuvent se chevaucher entre eux, nous vérifions que l'espace minimal occupé par ces composants est bien égal à l'aire du plus grand composant, c'est-à-dire l'espace libre à l'entrée (=62500).

Finalement, l'indice de faisabilité du problème d'agencement simple, exposé dans ce chapitre, est égal à 69 %, ce qui démontre à priori la faisabilité de ce problème d'agencement d'espace. En résumé, l'indice de faisabilité d'un problème d'agencement, présenté dans cette section, est une estimation de l'indice de faisabilité réel du problème qui, en plus d'être la solution optimale du problème d'optimisation défini dans l'équation 2.12, prend en compte la géométrie des composants. Cette estimation, toujours inférieure ou égale à l'indice de faisabilité réel du problème permet essentiellement de montrer au concepteur si le problème d'agencement ne peut pas être résolu, lorsque l'indice est supérieur à 100 %.

2.5 Conclusion

Ce chapitre a présenté quelques concepts innovants permettant de décrire et formuler de façon générique un problème d'optimisation d'agencement. Tout d'abord, la description des problèmes est basée sur la classification des composants en éléments matériels ou virtuels. Cette classification permet de créer un modèle géométrique du problème d'agencement, qui prend en compte, de manière générique, toutes les spécificités du problème. Cette classification des composants peut également

être plus détaillée en incluant les attributs de forme des composants (dimensions fixes ou variables et forme fixe ou variable).

De plus, la formulation du problème d'optimisation d'agencement utilisée dans la démarche, proposée dans ce manuscrit, est une formulation multiobjectif. Les choix du concepteur se font à posteriori sur les solutions de compromis générées par l'algorithme d'optimisation. Afin de formuler ce problème d'optimisation, un travail commun entre l'expert en optimisation et le concepteur est effectué, afin de traduire les exigences de ce dernier en variables d'optimisation, contraintes de conception et objectifs. Certaines de ces exigences sont difficilement traduisibles en fonctions mathématiques explicites. Il est donc nécessaire de proposer de nouvelles méthodes qui permettent de prendre en compte ces exigences dans la formulation du problème. Dans ce chapitre, le cas particulier de l'accessibilité aux composants depuis l'entrée du contenant est notamment développé. Deux approches, basées sur le polygone d'appartenance et sur l'optimisation de trajectoire sont proposées afin de traduire cette exigence d'accessibilité en contrainte ou objectif d'optimisation.

Enfin, ce chapitre présente un nouvel indice de faisabilité des problèmes d'agencement. Cet indice propose une alternative au traditionnel calcul de la compacité d'un problème d'agencement. Contrairement au calcul de la compacité habituellement utilisé, ce nouvel indice est basé sur la formulation d'une matrice d'intersections qui traduit les exigences de conception fixées par le concepteur. Cet indice permet de démontrer, de manière fiable, si le problème ne peut pas être résolu.

3 Stratégie d'optimisation modulaire

3.1 Introduction

Ce chapitre est dédié à l'étape de résolution proposée dans la démarche d'optimisation d'agencement présentée dans ce manuscrit. Cette étape de résolution consiste à trouver la ou les solutions optimales du problème d'optimisation d'agencement décrit et formulé dans les deux étapes précédentes. Cette étape de résolution consiste donc à utiliser une stratégie d'optimisation à la fois la plus générique et la plus efficace possible. En effet, l'objectif de cette stratégie d'optimisation est de pouvoir s'adapter au plus grand nombre de problèmes d'agencement et de résoudre ces problèmes avec un temps de calcul raisonnable.

Ce chapitre présente donc la stratégie d'optimisation modulaire qui a été développée au cours de ce doctorat. Cette stratégie d'optimisation s'appuie sur l'utilisation d'un algorithme génétique associé à différents modules d'optimisation participant à la recherche de solutions optimales. Dans ce chapitre, l'utilisation de l'algorithme génétique est tout d'abord justifié par la définition de la complexité d'un problème d'agencement. Ensuite, les différents modules et leur association avec l'algorithme génétique sont décrits.

3.2 Complexité d'un problème d'agencement

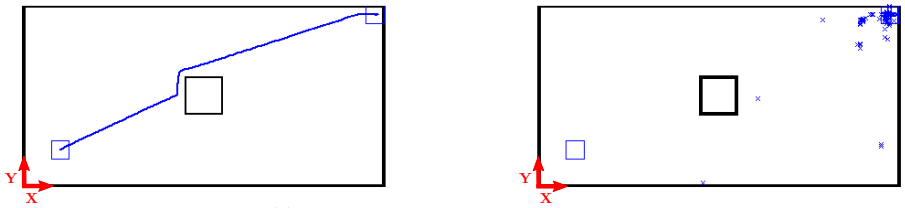
L'utilisation d'un algorithme stochastique, tel l'algorithme génétique, est justifiée par la grande complexité des problèmes d'agencement, notamment ceux issus d'applications industrielles. Il convient tout d'abord d'expliquer ce qu'est la complexité d'un problème d'agencement. La complexité de ces problèmes n'est pas clairement définie dans la littérature scientifique. Nous proposons d'éclaircir ce point en considérant cette complexité selon trois aspects :

- **complexité liée à la géométrie des composants** : lorsque les composants sont de forme irrégulière, des techniques particulières de calcul de collision doivent être implémentées. Aussi, cette détection des collisions consomme beaucoup de temps de calcul,
- **complexité liée à l'indice de faisabilité du problème (ou compacité)** : comme expliqué dans le chapitre précédent, l'indice de faisabilité rend compte de la difficulté de résolution du problème d'optimisation d'agencement. Plus cet indice est grand et plus la recherche de solutions admissibles, qui respectent toutes les contraintes de placement, est difficile. Le problème d'agencement est alors plus complexe,
- **complexité liée à la formulation du problème** : dans la plupart des problèmes d'optimisation d'agencement, les exigences du concepteur sont traduites en plusieurs contraintes et plusieurs objectifs. La multiplication de ces contraintes morcelle l'espace de conception, c'est-à-dire l'ensemble des combinaisons possibles des variables d'optimisation. Plus ces contraintes sont nombreuses et plus l'espace de conception est morcelé en zones séparées de solutions admissibles. Par conséquent, afin de passer d'une zone de solutions faisables à une autre, il est impossible d'utiliser les stratégies d'optimisation locale, notamment celles basées sur le calcul du gradient des contraintes. Les stratégies d'optimisation globale et stochastique, comme l'algorithme génétique, proposent donc une alternative efficace à la résolution de ces problèmes. Cependant, le temps de calcul augmente et le problème devient plus complexe.

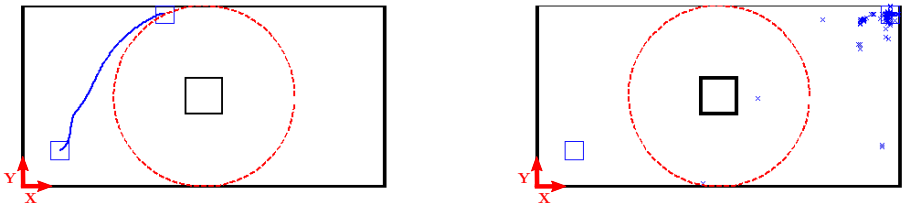
Ce dernier aspect est prépondérant dans la justification de l'utilisation d'un algorithme stochastique, tel l'algorithme génétique. Considérons par exemple un problème d'agencement simple en deux dimensions avec un contenant et deux composants carrés. Un des deux composants est fixe au milieu du contenant. Les variables d'optimisation sont les coordonnées du deuxième composant libre (x_2, y_2). Considérons ensuite 3 cas d'étude associés à ce problème d'agencement :

- cas n°1 : les contraintes de placement sont uniquement des contraintes géométriques de non-chevauchement entre le composant 2 libre et le composant 1 fixe et des contraintes d'appartenance du composant 2 au contenant. Le critère d'optimisation est de maximiser la distance entre le composant 2 et l'origine du repère correspondant au coin en bas à gauche du contenant,
- cas n°2 : on ajoute au cas n°1 une contrainte fonctionnelle : $d_{12} = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2} > D$. Cette contrainte signifie que la distance entre les composants 1 et 2 doit être supérieure à un certaine distance arbitrairement choisie,
- cas n°3 : on ajoute au cas n°2 une nouvelle contrainte fonctionnelle : $\mid y_1 - y_2 \mid > h$, où le paramètre h est également arbitrairement choisi.

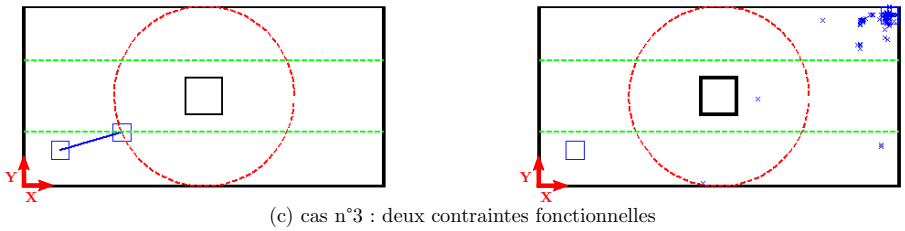
La figure 3.1 illustre la résolution des trois cas d'étude via d'une part, l'utilisation d'un algorithme de programmation quadratique séquentielle (SQP) basé sur le calcul du gradient, et d'autre part, l'utilisation d'un algorithme génétique. La partie gauche de la figure montre les solutions trouvées par l'algorithme SQP (chemin parcourue par la solution courante) et la partie droite montre les solutions trouvées par l'algorithme génétique (individus générés).



(a) cas n°1 : pas de contrainte fonctionnelle



(b) cas n°2 : une contrainte fonctionnelle



(c) cas n°3 : deux contraintes fonctionnelles

FIGURE 3.1 – *Comparaison de l'algorithme SQP et de l'algorithme génétique sur un problème d'agencement simple*

Pour le premier cas d'étude, sans contrainte fonctionnelle, l'espace de conception n'est pas morcelé. Quel que soit le point initial, l'algorithme SQP parvient à trouver la solution optimale. Pour le second cas d'étude, l'espace des solutions admissibles est divisé en deux parties. Pour le troisième cas d'étude, il est divisé en quatre parties. Ainsi, pour ces deux derniers cas d'étude, si le point initial ne se trouve pas dans la zone admissible de l'espace de conception, où se trouve également la solution optimale, l'algorithme ne peut pas faire tendre la solution courante vers la solution optimale. Quelle que soit la population initiale, l'algorithme génétique parvient lui à diriger sa recherche vers la zone de solutions où se trouve la solution optimale.

Finalement, pour un problème d'optimisation d'agencement simple, on remarque que les algorithmes traditionnels, basés sur le calcul du gradient, ne sont pas adaptés aux cas où l'espace de conception est morcelé dû à la multiplication des contraintes de conception. La complexité des problèmes d'agencement, notamment celle liée à la formulation des problèmes, conduit donc à utiliser des algorithmes d'optimisation stochastiques. La stratégie d'optimisation, présentée dans ce chapitre, est basée sur l'utilisation d'un algorithme stochastique : l'algorithme génétique. Le principe de

fonctionnement de cet algorithme est notamment décrit dans le chapitre 1 de ce manuscrit.

3.3 Modules d'optimisation

Les modules d'optimisation, présentés dans cette section, ont pour objectif d'aider l'algorithme génétique dans sa recherche de solutions optimales. En effet, dans un des cas d'application résolus dans le chapitre 5, l'algorithme génétique seul n'a pas pu trouver de solution admissible, qui respecte toutes les contraintes du problème d'optimisation. Ceci s'explique notamment par la forte complexité du problème. Cette partie présente quatre modules différents. Certains de ces modules sont spécifiques à certains types de problème d'agencement. L'association de ces modules avec l'algorithme génétique forme la stratégie modulaire utilisée dans la méthode de conception présentée dans ce manuscrit.

Les modules d'optimisation ne sont pas liés entre eux et peuvent être utilisés seuls ou associés à d'autres modules. Il est bien entendu possible d'ajouter d'autres modules à cette stratégie modulaire, si ces modules apportent leur contribution à la recherche de solutions. Dans cette partie, les modules d'optimisation qui sont décrits sont identifiés par des acronymes qui seront utilisés dans la suite de ce document pour faire référence à ces mêmes modules.

3.3.1 Module « OPEA » : Optimisation du Placement des Espaces d'Accessibilité

Le chapitre 2 a montré que les composants d'un problème d'agencement peuvent être classés en deux catégories : les composants matériels et les composants virtuels. Un composant virtuel peut notamment être utilisé pour modéliser l'espace d'accessibilité d'un composant matériel, c'est-à-dire l'espace libre placé près du composant matériel et nécessaire au bon fonctionnement de ce dernier. Ce module d'optimisation est dédié à l'optimisation du placement de ces espaces d'accessibilité.

Pour mieux comprendre le principe de fonctionnement de ce module d'optimisation, considérons le placement en deux dimensions de deux composants matériels, de forme rectangulaire. Le placement de ces deux composants est piloté par des variables d'optimisation continues (X,Y) qui caractérisent le centre géométrique de chaque composant et une variable discrète α qui définit l'orientation de chaque composant (0° ou 90°). Supposons également que le composant matériel n°1 requiert un espace d'accessibilité. Pour chaque orientation α du composant n°1, il existe donc deux positions possibles pour placer cet espace d'accessibilité. Ces deux possibilités sont donc modélisées par une variable d'optimisation discrète λ qui peut prendre deux valeurs (par exemple 1 ou 2). Cette variable λ caractérise le sens du composant. La figure 3.2, illustre les deux choix possibles de placement de cet espace d'accessibilité pour une orientation donnée des deux composants matériels.

L'objectif de ce module d'optimisation est donc d'optimiser le placement de cet espace d'accessibilité. Ainsi, la variable d'optimisation λ n'est plus prise en compte et l'espace d'accessibilité est

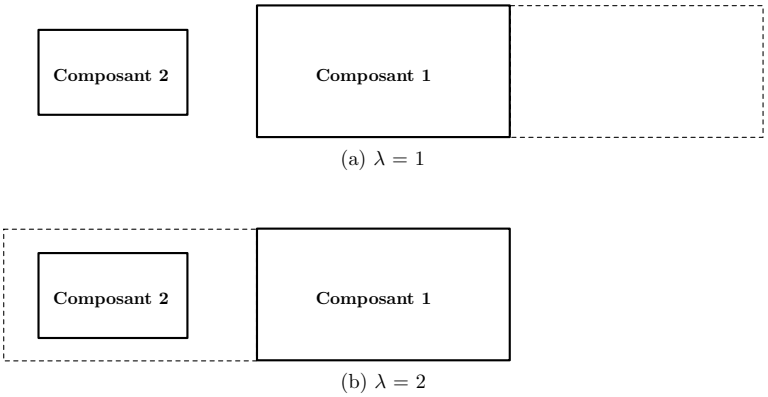


FIGURE 3.2 – *Module OPEA : Optimisation du Placement des Espaces d'Accessibilité*

positionné de façon à minimiser les contraintes de placement entre celui-ci et les autres composants de l'agencement, selon les exigences exprimées par le concepteur. Le nombre de variables d'optimisation diminue et la recherche de solutions admissibles est facilitée. Dans l'exemple illustré sur la figure 3.2, le module d'optimisation OPEA choisit le cas (a), avec $\lambda = 1$, car le chevauchement entre l'espace d'accessibilité du composant n°1 et le composant n°2 est minimisé. Dans un agencement, lorsqu'il existe plusieurs espaces d'accessibilité, le module OPEA traite les composants un par un dans l'ordre dans lequel ils ont été définis par le concepteur. Le placement optimal des espaces d'accessibilité est donc séquentiel.

La figure 3.3 montre comment connecter ce module d'optimisation à l'algorithme génétique. Avant d'évaluer les individus d'une population, le module d'optimisation optimise le placement de tous les espaces d'accessibilité présents dans l'agencement.

3.3.2 Module « SEPA » : SEPAration des composants

Ce module d'optimisation vise à aider l'algorithme génétique à générer des solutions qui respectent toutes les contraintes du problème d'optimisation, notamment les contraintes de non-chevauchement entre composants. En effet, il réalise, pour chaque individu généré par l'algorithme génétique, une optimisation locale qui résout le problème d'optimisation mono-objectif sans contrainte suivant :

$$\text{Problème P} : \begin{cases} \text{trouver } \mathbf{x}^* = (x_1, x_2, ..., x_n) \\ \mathbf{x}^* = \text{argmin}\,(F(\mathbf{x})) \end{cases} \tag{3.1}$$

Ce module de séparation agit sur les variables d'optimisation continues qui pilotent le placement des centres géométriques des composants (n variables). La fonction F caractérise le respect des contraintes de conception, notamment le non-chevauchement des composants entre eux, l'appartenance des composants au contenant et les autres contraintes du problèmes d'agencement. Il existe différentes méthodes pour exprimer cette fonction F. La formulation choisie ici est basée sur les

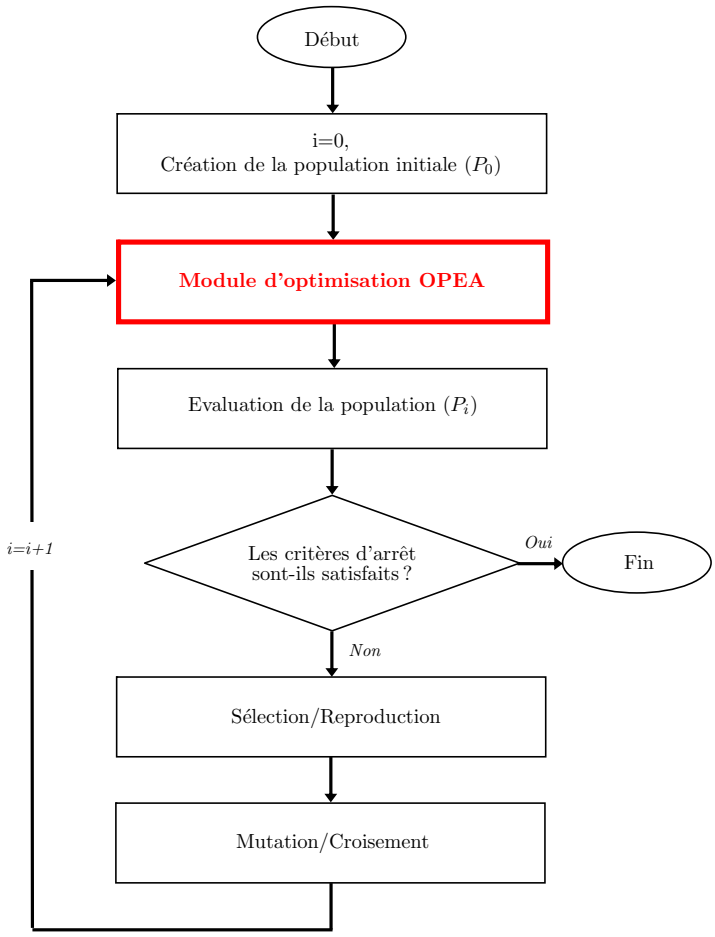


FIGURE 3.3 – *Algorithme génétique couplé avec le module d'optimisation OPEA*

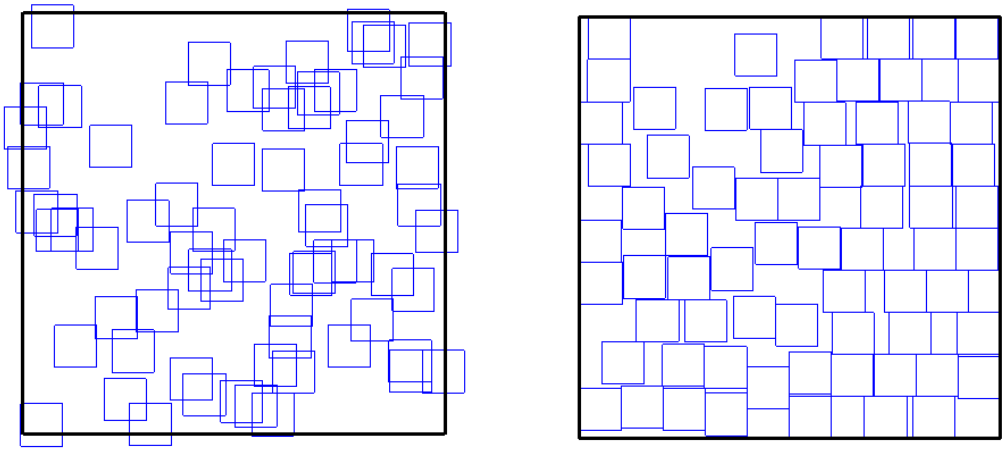
recherches menées dans (Jac10) :

$$F(\mathbf{x}) = w_1 \times f_1(\mathbf{x}) + w_2 \times f_2(\mathbf{x}) + w_3 \times f_3(\mathbf{x}), \text{ avec } w_1 + w_2 + w_3 = 1 \tag{3.2}$$

La fonction f_1 traduit le chevauchement des composants entre eux, la fonction f_2 traduit l'appartenance des composants au contenant et la fonction f_3 traduit les autres contraintes du problème d'optimisation. Les coefficients w_1, w_2 et w_3 définissent des paramètres permettant de régler l'influence de chacune des fonctions sur le comportement global de la fonction objectif de ce problème de séparation. La fonction f_1, calculée pour deux composants, est égale au produit de l'aire (ou volume pour les problèmes en trois dimensions) d'intersection entre ces deux composants par la profondeur de pénétration d'un composant dans l'autre. De la même manière, le fonction f_2, calculée pour un composant, est égale au produit de l'aire (ou volume pour les problèmes en trois dimensions) de la partie du composant située à l'extérieure du contenant par la distance modélisant la translation nécessaire pour que le composant soit dans le contenant. L'expression analytique de ces deux fonctions et de leurs gradients respectifs est détaillée dans (Jac10).

L'algorithme d'optimisation utilisé pour exécuter ce module de séparation des composants est basé sur la méthode quasi-Newton BFGS (Bro70; Fle70; Gol70; Sha70). Il existe, pour cette méthode, trois critères d'arrêt : le nombre maximal d'itérations, la précision sur la norme du gradient du critère d'optimisation et un seuil au-dessous duquel l'amélioration de la valeur de l'objectif n'est plus suffisante. Afin d'éviter l'utilisation de la différentiation numérique, l'algorithme d'optimisation a recours au gradient analytique de la fonction objectif.

La figure 3.4 illustre le fonctionnement de ce module d'optimisation SEPA pour un problème d'agencement simple en deux dimensions. Le contenant est un carré de côté égal à 100 unités de longueur. Chaque composant, placé dans ce contenant, est également un carré de côté égal à 10. Le module SEPA consiste donc à positionner tous les composants dans le contenant, en minimisant le chevauchement entre ces composants. Sur la figure 3.4, la compacité du problème est égale à 70 %. Le tableau 3.1 récapitule les résultats du test du module d'optimisation SEPA sur ce problème d'agencement : la valeur de la fonction F, qui caractérise le respect des contraintes de placement, pour les solutions initiale et optimisée et le nombre d'itérations nécessaires à l'algorithme pour minimiser F.



Solution initiale Solution optimisée

FIGURE 3.4 – *Fonctionnement du module SEPA pour un problème d'agencement 2D*

Compacité	F (valeur initiale)	F (valeur finale)	Nombre d'itérations
70 %	$1{,}13.10^4$	0,07	67

TABLE 3.1 – *Test du module SEPA pour un problème d'agencement 2D*

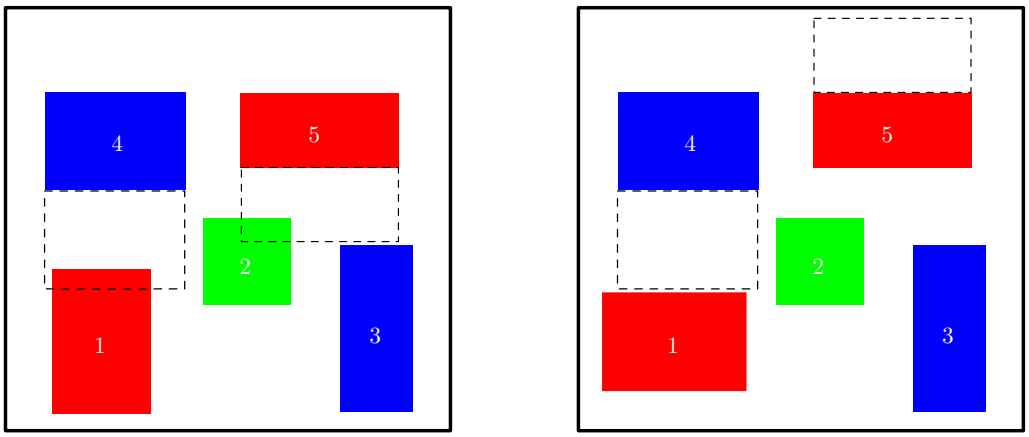
Ce module de séparation peut être intégré dans l'algorithme génétique de manière à faciliter la recherche de solutions admissibles, qui respectent les contraintes du problème d'optimisation d'agencement. L'implémentation de ce module SEPA dans l'algorithme génétique conduit à la création d'un algorithme hybride, présenté dans la chapitre 1 et plus largement détaillé dans (Jac10). La structure de cet algorithme hybride est similaire à la structure de l'algorithme, illustré sur la figure 3.3, où le module d'optimisation OPEA est remplacé par le module d'optimisation SEPA.

3.3.3 Module « PERT » : PERTurbation locale

Le module d'optimisation PERT a pour objectif de perturber localement l'agencement courant. Il vise à modifier de manière aléatoire l'orientation et le sens de certains composants. L'utilisation de ce module PERT est très judicieuse lorsque qu'elle est liée à l'utilisation du module d'optimisation SEPA. En effet, le module SEPA, qui sépare les composants afin de respecter au mieux les contraintes de non-chevauchement et les autres contraintes du problème, modifie uniquement les variables continues de positionnement des centres géométriques des différents composants. Le module d'optimisation SEPA n'a aucune influence sur les variables discrètes, notamment les variables d'orientation et de sens des composants. Ceci signifie donc que pour un agencement donné, il se peut que l'algorithme d'optimisation utilisé par le module SEPA trouve un minimum local qui minimise les contraintes du problème d'optimisation.

L'objectif de ce module d'optimisation PERT est donc ici d'empêcher le module de séparation des composants de générer une solution issue d'un minimum local, qui ne respecte pas toutes les contraintes de conception. L'idée est alors d'appliquer une perturbation locale sur les variables d'orientation et de sens des composants, si la solution trouvée par l'algorithme de séparation des composants ne respecte pas toutes les contraintes. Cette perturbation locale consiste à choisir aléatoirement k composants (k fixé arbitrairement) et modifier l'orientation et/ou le sens des ces composants, également de manière aléatoire. Après avoir perturbé localement l'agencement, le module de séparation des composants peut être ré-exécuté de manière à minimiser les contraintes du problème d'optimisation d'agencement.

La figure 3.5 illustre le fonctionnement du module PERT pour un problème d'agencement simple en deux dimensions. Les composants 1 et 5 ont changé, respectivement, d'orientation et de sens.



Solution initiale Solution modifiée

FIGURE 3.5 – *Fonctionnement du module PERT pour un problème d'agencement 2D*

La figure 3.6 illustre l'algorithme qui résulte de la combinaison de l'algorithme génétique, du module OPEA qui optimise le placement des espaces d'accessibilité, du module SEPA de séparation des

composants et du module PERT qui applique une perturbation locale.

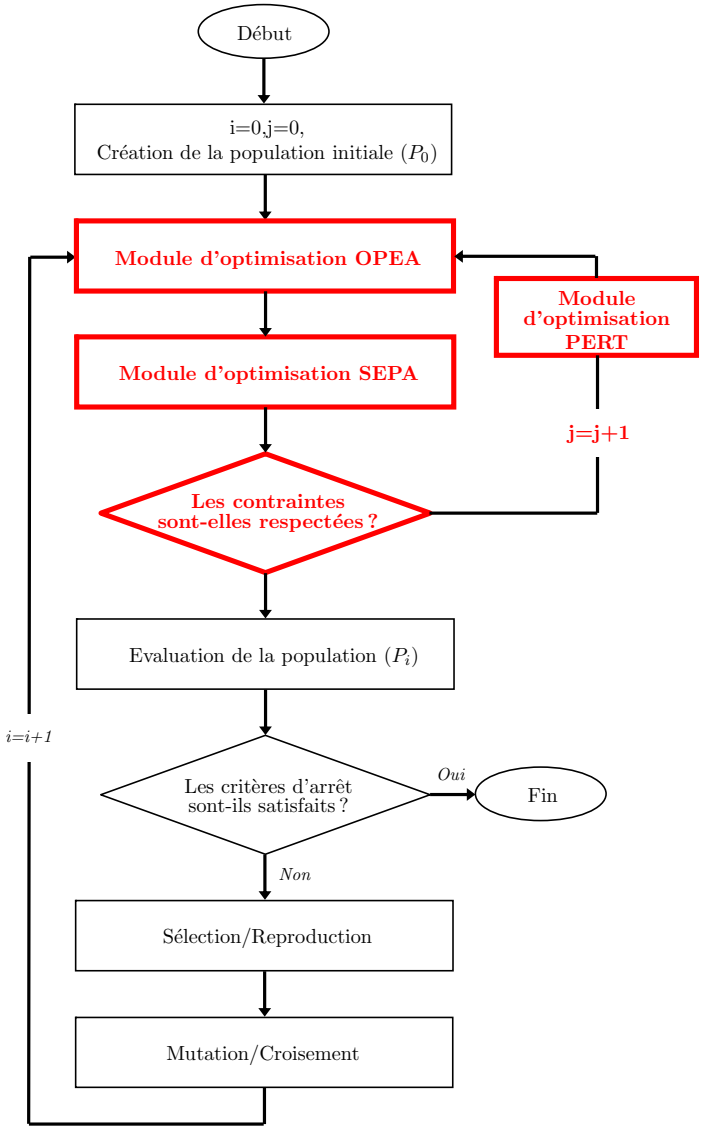


FIGURE 3.6 – *Algorithme génétique couplé avec les modules d'optimisation OPEA, SEPA et PERT*

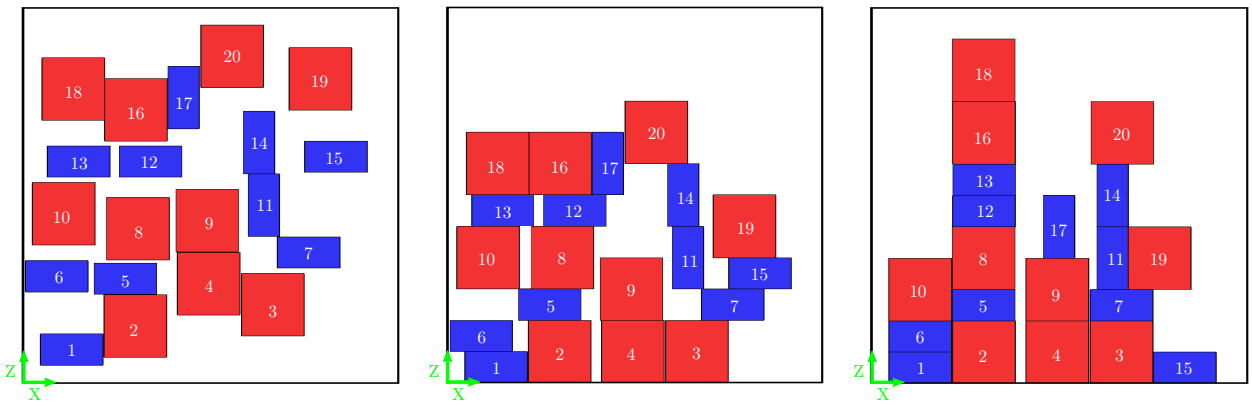
Il existe un paramètre de réglage de la perturbation locale : j_{max}. Ce paramètre définit, pour chaque individu traité par l'algorithme génétique, le nombre maximal de fois que la perturbation locale pourra être appliquée si la solution courante ne respecte pas toutes les contraintes de conception. Aussi, si la solution courante n'est pas admissible et que j_{max} est atteint, l'algorithme génétique prend en compte la solution qui minimise les contraintes parmi toutes celles générées précédemment, et continue son processus de génération d'individus.

3.3.4 Module « GRAV » : prise en compte de la GRAVité

L'objectif du module d'optimisation GRAV est de prendre en compte la loi de la gravité pour les problèmes d'optimisation d'agencement en trois dimensions. En effet, dans les applications d'agencement d'espace en trois dimensions, notamment dans les problématiques de stockage de composants, de nombreux algorithmes d'optimisation ont été développés. Ils consistent pour la plupart d'entre eux à poser l'un après l'autre les composants, assurant ainsi le respect de la loi de la gravité dans l'agencement. L'utilisation de la stratégie modulaire, présentée ici, et basée sur l'utilisation de l'algorithme génétique, ne garantit pas seule le respect de la gravité. Il est donc nécessaire d'associer à l'algorithme génétique un module d'optimisation qui garantisse le fait que tous les composants soient posés sur un support (sol ou autre composant).

Pour comprendre le fonctionnement du module GRAV, considérons un problème d'agencement simple, en deux dimensions. Ce problème consiste à placer 20 composants (2 types de composants) dans un contenant, tout en respectant la contrainte de gravité. Ce problème d'agencement est illustré sur la figure 3.7(a), où la gravité n'est pas encore prise en compte. On suppose que la gravité s'applique suivant l'axe $(O, \overrightarrow{Z})$. Ce module d'optimisation GRAV consiste tout d'abord à classer les composants suivant leur position suivant l'axe $(O, \overrightarrow{Z})$, c'est-à-dire du plus proche du sol ($Z = 0$) au plus éloigné. Le numéro indiqué sur chaque composant, sur la figure 3.7, corresponde à la place de ce composant dans ce classement. Ensuite, pour chaque composant, en partant du plus proche du sol, le module GRAV détecte s'il existe un autre composant placé en dessous de celui-ci. S'il n'en existe pas, le composant considéré est placé sur le sol, sinon, il est positionné sur le composant placé en dessous et ayant la position suivant l'axe $(O, \overrightarrow{Z})$ la plus élevée. La figure 3.7(b) illustre ce fonctionnement pour les 20 composants de l'agencement. Par exemple, lorsque le module GRAV traite le composant 8, il détecte que les composants 2 et 5 sont placés en dessous. Le composant 5 a la position la plus élevée suivant l'axe $(O, \overrightarrow{Z})$, par conséquent le composant 8 est positionné sur le composant 5. Afin de détecter qu'un composant est placé en dessous d'un autre composant, le module GRAV calcule l'aire d'intersection entre les surfaces de projection des deux composants suivant un plan normal à l'axe d'application de la gravité (axe $(O, \overrightarrow{Z})$ sur la figure 3.7). Si cette aire d'intersection est non nulle, ceci signifie alors que un des composants est placé en dessous de l'autre.

Sur la figure 3.7(b), on remarque que certains composants ne sont pas positionnés correctement sur le composant placé en dessous (par exemple, le composant 10 sur le composant 5). L'agencement ne serait pas réalisable car le composant 10 tomberait sous l'influence de la gravité. Afin d'atténuer ce problème de mauvais positionnement, il est possible d'ajouter au module GRAV une nouvelle fonctionnalité. En effet, lorsqu'un composant est positionné sur un autre composant, une optimisation locale, mono-objectif et sous contraintes, est réalisée. Cette optimisation locale vise à maximiser la surface de contact entre ces deux composants (aire d'intersection des deux surfaces projetées des deux composants), tout en respectant les contraintes de non-chevauchement avec les autres composants. La figure 3.7(c) illustre le fonctionnement de ce module GRAV amélioré pour le problème



(a) Agencement initial

(b) Prise en compte de la gravité

(c) Prise en compte de la gravité et surface de contact optimale

FIGURE 3.7 – *Module GRAV : prise en compte de la GRAVité*

d'agencement pris pour exemple.

Comme pour les autres modules d'optimisation, il est possible également d'intégrer le module GRAV dans la structure globale de l'algorithme génétique. La figure 3.8 illustre l'algorithme qui résulte de la combinaison de tous les modules d'optimisation présentés dans ce chapitre.

3.4 Conclusion

Ce chapitre 3 a détaillé la stratégie d'optimisation modulaire utilisée dans la démarche d'optimisation proposée dans ce manuscrit. Cette stratégie d'optimisation est basée sur l'utilisation d'un algorithme génétique couplé avec des modules d'optimisation locale. L'utilisation de l'algorithme génétique est justifiée par la grande complexité de la plupart des problèmes d'agencement d'espace. Dans ce chapitre, la complexité des problèmes d'optimisation d'agencement est définie. Elle est basée sur trois critères : la géométrie des composants, l'indice de faisabilité du problème et le nombre de contraintes de conception. Le troisième critère caractérise le degré de morcellement de l'espace de conception qui induit nécessairement l'utilisation d'un algorithme stochastique tel l'algorithme génétique.

De plus, pour aider l'algorithme génétique dans sa recherche de solutions optimales et pour s'adapter au mieux aux spécificités de certains problèmes d'agencement, des modules d'optimisation sont associés à l'algorithme génétique pour former la stratégie d'optimisation modulaire présentée dans ce chapitre. Chaque module d'optimisation est indépendant, et peut être associé à tout autre module, ce qui permet au concepteur d'adapter la stratégie d'optimisation modulaire aux spécificités de son problème d'agencement et à ses propres exigences liées à la génération de solutions. En effet, parce que l'utilisation de ces modules augmentent légèrement le temps de calcul, un compromis est à faire entre le temps de calcul alloué au processus d'optimisation et le nombre solutions optimales obtenues.

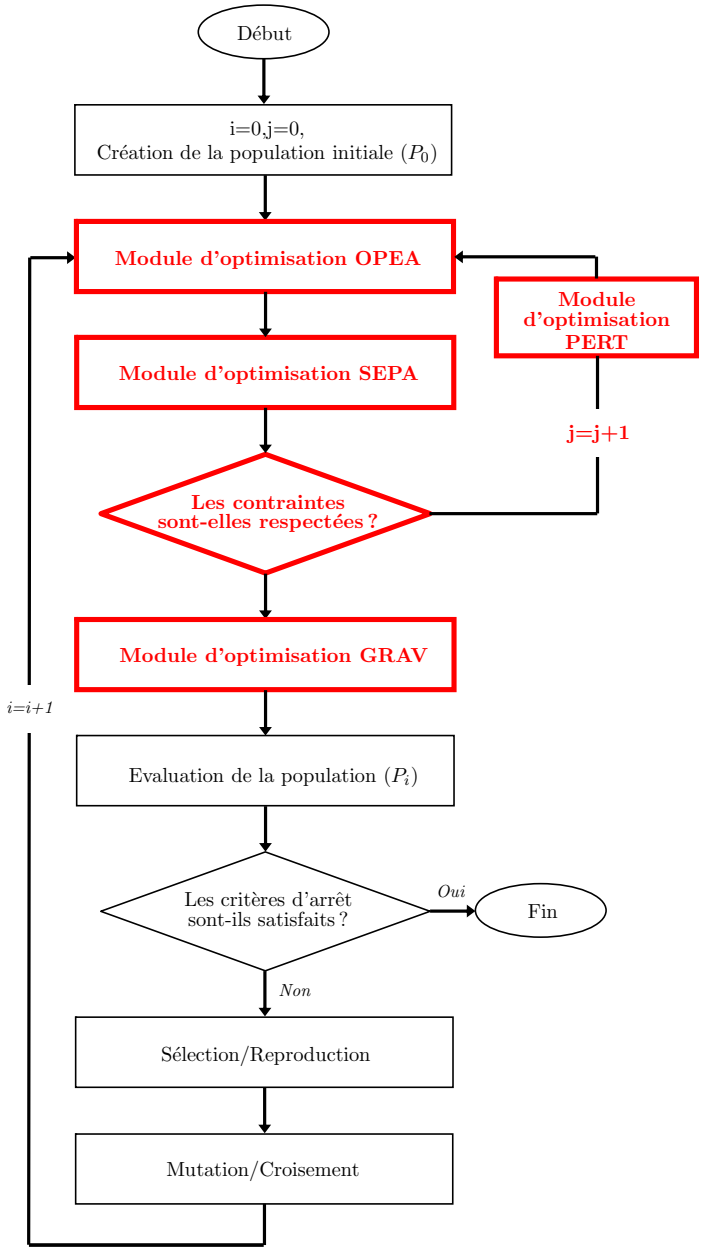


FIGURE 3.8 – *Algorithme génétique couplé avec les modules d'optimisation OPEA, SEPA, PERT et GRAV*

Enfin, différents algorithmes d'optimisation découlent de cette stratégie d'optimisation modulaire, en associant un ou plusieurs modules à l'algorithme génétique. Plusieurs de ces algorithmes sont implémentés et testés sur un cas d'application présenté dans le chapitre 5 et leurs résultats sont comparés quantitativement et qualitativement. L'étude comparative de ces algorithmes est également décrite dans (BPBR11c; BPBR11b).

4 Interactivité et prise de décision

4.1 Introduction

Le concepteur joue un rôle essentiel dans un processus d'optimisation de conception, que ce soit dans la formulation du problème d'optimisation, l'interactivité avec l'algorithme ou la prise de décision finale. La démarche d'optimisation, proposée dans ce manuscrit, se focalise essentiellement sur la dernière étape qui consiste, pour le concepteur, à faire des choix de conception, sur les différentes alternatives générées par la stratégie d'optimisation. En effet, comme expliqué au chapitre 2, la formulation adoptée pour écrire le problème d'optimisation d'agencement est une formulation mutliobjectif. Cette formulation conduit donc à la génération d'un ensemble de solutions qui réalisent le meilleur compromis entre tous les critères d'optimisation. Il est donc nécessaire de proposer au concepteur un ensemble de méthodes et d'outils lui permettant de choisir la solution qui convienne le mieux à son expertise personnelle et son propre jugement du problème d'agencement.

Les méthodes, détaillées dans ce chapitre, consistent premièrement à trier les solutions générées par l'algorithme d'optimisation. La notion de « variante » est notamment définie. Ensuite, la prise de décision finale est basée sur une visualisation multidimensionnelle de solutions et une comparaison

des performances de celles-ci. Aussi, une stratégie innovante, permettant d'explorer un ensemble de solutions de manière perceptive, est décrite dans ce chapitre. Cette stratégie permet notamment d'intégrer aux solutions optimales, les exigences du concepteur liées aux aspects qualitatifs et perceptifs, qui ne peuvent pas facilement être pris en compte dans la formulation du problème d'agencement. Enfin, la dernière partie de ce chapitre est dédiée à l'interaction entre le concepteur et une solution proposée par l'algorithme. L'objectif de cette interactivité est d'améliorer manuellement et localement une solution en prenant en compte le jugement personnel et l'expertise du ou des concepteurs qui travaillent ensemble sur le problème d'optimisation d'agencement. Dans cette partie, une interface de réalité virtuelle, particulièrement adaptée aux problèmes d'agencement en trois dimensions, est implémentée dans l'interface interactive de représentation d'une solution.

4.2 Tri et exploration de solutions d'agencement

La formulation multiobjectif du problème d'optimisation d'agencement permet de générer un ensemble de solutions optimales, réalisant le meilleur compromis entre tous les critères d'optimisation. L'objectif est donc dans un premier temps de proposer une méthode et des outils permettant de trier, visualiser et explorer cet ensemble de solutions.

4.2.1 Définition d'une variante admissible

Premièrement, il convient de détecter, parmi toutes le solutions générées par l'algorithme d'optimisation, les solutions qui respectent les contraintes de conception. Il est également possible de relaxer certaines contraintes, en fixant arbitrairement un seuil de respect de la contrainte. Finalement, une solution Sol qui respecte les contraintes est appelée solution « admissible » ou « faisable » et respecte la condition suivante :

$$\begin{cases} \forall\, i \in \{1, ..., n_c\} \\ C_i(Sol) \leqslant \triangle c_i \end{cases} \tag{4.1}$$

où $C_i(Sol)$ définit la $i^{ème}$ contrainte pour la solution Sol. $\triangle c_i$ définit le seuil de respect des contraintes pour la $i^{ème}$ contrainte et n_c définit le nombre total de contraintes de conception du problème.

Ensuite, dans la plupart des problèmes, les variables de positionnement des composants sont définies par des variables continues, et l'algorithme d'optimisation propose donc des solutions très similaires où par exemple, entre deux solutions, seulement un des composants a été déplacé de ε unités de mesure, avec ε très petit. Pour le concepteur, il est donc difficile de faire la différence entre ces deux solutions et il est inutile de les visualiser toutes les deux. Ainsi, la méthode de conception, décrite dans ce manuscrit, propose de définir la notion de variante, c'est-à-dire la notion de solution

géométriquement et fonctionnellement différente. Deux solutions Sol_i et Sol_j sont deux variantes, si elles diffèrent par l'un des trois critères suivants :

- un des composants de l'agencement a été déplacé d'au moins Δ_g unités de mesure, Δ_g fixé arbitrairement par le concepteur (Δ_g est le seuil de différenciation géométrique),
- un des composants a changé d'orientation ou de sens,
- la différence maximale entre les valeurs des objectifs pour les deux solutions est supérieur à Δ_f, fixé arbitrairement par le concepteur (Δ_f est le seuil de différenciation fonctionnelle).

4.2.2 Visualisation multidimensionnelle de solutions

Lorsque les solutions, générées par l'algorithme, ont été triées, le concepteur a besoin de comparer ces solutions. Il convient donc d'afficher, sur une même interface graphique, toutes les informations utiles à la sélection d'une solution idéale (objectifs, contraintes, variables, modèle géométrique...). Les deux outils présentés dans les parties suivantes, permettent cet affichage multidimensionnelle des solutions et s'inspirent de l'approche « *design by shopping* » (Bal99), évoquée dans l'état de l'art de ce manuscrit.

4.2.2.1 Nuage de points

Le nuage de points est adapté à la représentation de solutions en une, deux voire trois dimensions. Le nuage de points consiste à représenter dans un repère orthogonal les solutions d'agencement par des points qui ont comme coordonnées certaines données numériques de l'agencement, choisies par l'utilisateur (par exemple, la valeur des objectifs, des contraintes ou de certaines variables). Ce nuage de points peut notamment être restreint aux solutions optimales du problème d'optimisation d'agencement et associer à chaque solution la valeur des critères d'optimisation. La nuage de points forme alors la frontière de Pareto du problème d'optimisation et permet de comparer les solutions selon leurs objectifs.

La figure 4.1 illustre l'interface graphique permettant de visualiser un nuage de points de solutions en une, deux ou trois dimensions. L'exemple considéré dans la figure 4.1 fait référence au problème d'optimisation d'agencement détaillé dans le chapitre 5. Les solutions sont ici comparées selon la valeur de leurs deux objectifs. L'interface graphique, représentée par la figure 4.1 fait partie des outils présents dans un démonstrateur d'optimisation d'agencement, développé dans le cadre de ce doctorat. Ce démonstrateur est notamment décrit dans la dernière partie du chapitre 5.

La particularité de l'outil, présenté sur la figure 4.1, réside dans son interaction avec le concepteur. En effet, le nuage de points est d'abord un outil de visualisation et de comparaison des données représentant les agencements considérés. L'interface intègre en plus ici un outil de visualisation du modèle géométrique d'une solution d'agencement, sélectionné par le concepteur. Lorsque ce dernier clique sur un point, représentant une solution, le modèle géométrique ainsi que les valeurs des

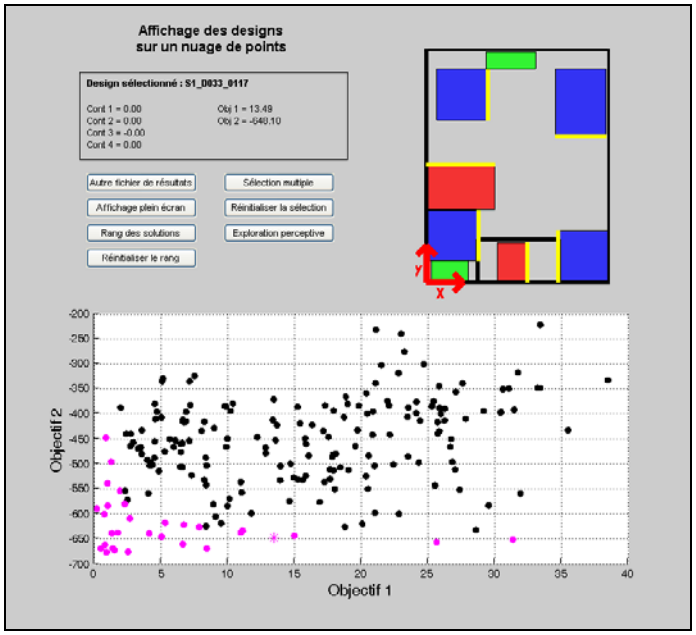


FIGURE 4.1 – *Nuage de points de solutions en deux dimensions*

données considérées dans la construction du nuage de points s'affichent.

Aussi, lorsque le concepteur utilise le nuage de points pour représenter les solutions selon la valeur de leurs objectifs, il est possible de mettre en surbrillance et de sélectionner les solutions qui ont le même rang (rang de Fonseca/Fleming ou rang de Goldberg, définis dans le chapitre 1). La figure 4.1 illustre cette fonctionnalité. Les solutions, qui ont un rang (de Goldberg) compris entre 1 et 5 sont mis en surbrillance. Les solutions, sur le front de Pareto, ont un rang égal à 1. Enfin, ce nuage de points intègre un outil de sélection multiple, qui permet de sélectionner des solutions. Ensuite, l'ensemble des solutions sélectionnées par le concepteur peut par exemple être exploré de manière perceptive via la stratégie détaillée dans la suite de ce chapitre.

4.2.2.2 Graphe parallèle

Le graphe parallèle est un autre outil de visualisation et de comparaison de solutions. La différence qui existe entre cet outil et le nuage de points réside dans la possibilité de visualiser et comparer les solutions en n_d dimensions, où n_d représente le nombre de données (variables, contraintes ou objectifs) prises en compte par le concepteur. Ce paramètre n_d, à l'inverse du nuage de points, peut être supérieur à 3. Le principe général du graphe parallèle est de représenter chaque donnée sur un axe vertical. Les bornes inférieures et supérieures de chaque axe vertical dépend des minima et maxima des données considérées par le concepteur, sur l'ensemble des solutions à comparer.

Ensuite, chaque solution est représentée par une ligne brisée et cette ligne brisée coupe chaque axe vertical en un point ayant pour coordonnée la valeur de la donnée correspondante pour la solution

considérée. La figure 4.2 illustre ce graphe parallèle pour le problème d'optimisation d'agencement, détaillé dans le chapitre 5. Comme pour le nuage de points, le graphe parallèle est également implémenté dans le démonstrateur d'optimisation d'agencement.

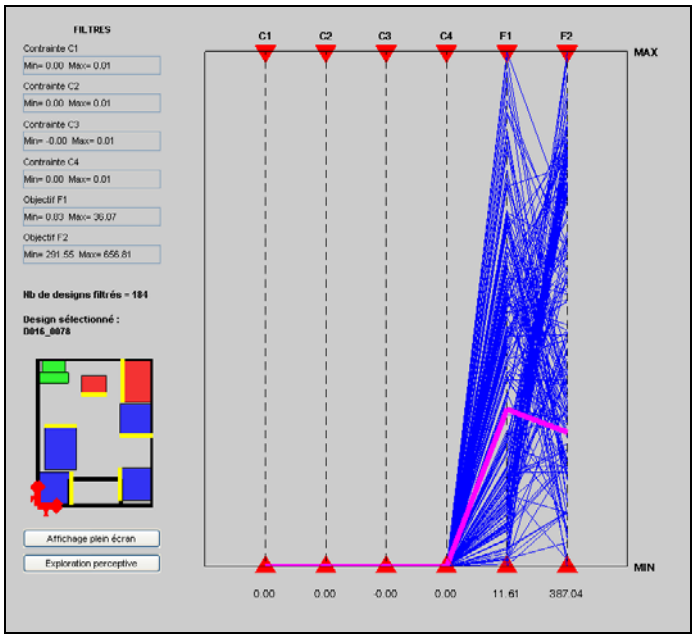


FIGURE 4.2 – *Graphe parallèle pour trois contraintes et trois objectifs*

Ce graphe parallèle est également un outil interactif car le concepteur peut visualiser le modèle géométrique et la valeur des données correspondant à une solution, sélectionnée par un clique sur une ligne brisée. L'autre particularité de cet outil est de pouvoir filtrer certaines solutions, en pilotant les bornes des axes verticaux représentant les données considérées par le concepteur. Si on considère par exemple, que les données prises en compte par le concepteur font référence aux objectifs du problème d'optimisation, le concepteur peut favoriser certaines solutions en fonction de ses propres références sur les critères d'optimisation. Aussi, via l'utilisation des filtres, le concepteur peut analyser l'influence des données entre elles, par exemple l'influence de certaines variables sur les performances de l'agencement. Les solutions filtrées par le concepteur peuvent ensuite faire l'objet d'une exploration perceptive.

4.2.3 Exploration perceptive de solutions

4.2.3.1 Introduction

Cette partie s'intéresse plus particulièrement à la prise en compte des perceptions du concepteur, dans l'exploration de solutions d'agencement générées par l'algorithme d'optimisation. En effet, les outils de visualisation multidimensionnelle, présentés dans la partie précédente permettent de visualiser les solutions en prenant uniquement en compte des critères quantitatifs (variables, contraintes

ou objectifs). Cependant, d'autres critères, plus subjectifs, peuvent être pris en considération dans le choix d'une solution finale. Ces critères subjectifs, font généralement référence aux exigences qui ont été exprimées par le concepteur mais qui n'ont pu être intégrées dans la formulation du problème d'optimisation d'agencement. Ces exigences peuvent faire référence à l'expertise du concepteur, à ses connaissances métier ou encore à ses propres critères perceptifs par rapport au problème d'agencement. Ces exigences sont difficiles à être formulées par une expression mathématique simple. La méthode proposée ici consiste donc à explorer un ensemble de solutions (solutions générées par l'algorithme d'optimisation, triées et comparées quantitativement), en prenant en compte la perception du concepteur sur le problème d'agencement. Cette méthode est basée sur l'utilisation d'un algorithme génétique interactif (IGA), qui est décrit dans l'état de l'art de ce manuscrit.

4.2.3.2 Correspondance entre espace de conception et espace à explorer

L'algorithme génétique interactif est un algorithme évolutionnaire, basé sur la génération de populations d'individus. Chaque individu représente ici un agencement. Il existe donc un espace de conception dans lequel l'algorithme génétique interactif génère des solutions. Cet espace de conception est défini par l'ensemble des combinaisons possibles entre toutes les variables (continues ou discrètes) du problème. La difficulté majeure de la méthode proposée repose sur le fait que l'espace des solutions à explorer ne représente qu'une partie de cet espace de conception de l'IGA.

Le mécanisme de correspondance, décrit dans ce paragraphe, consiste donc à faire correspondre à chaque solution de l'espace de conception de l'IGA une solution de l'ensemble des agencements à explorer perceptivement. Pour faire correspondre un individu a de l'espace de conception de l'IGA à un individu b de l'espace à explorer, la distance d_{a-b} est calculée. Cette distance peut être définie suivant différentes normes :

- norme 1 : $d_{a-b} = \sum_{i=1}^{n} |X_a(i) - X_b(i)|$
- norme 2 (ou norme euclidienne) : $d_{a-b} = \sqrt{\sum_{i=1}^{n} (X_a(i) - X_b(i))^2}$
- norme infinie : $d_{a-b} = \max_{i=1}^{n} |X_a(i) - X_b(i)|$

où X_a définit le vecteur des variables de conception pour l'agencement a et X_b, ce même vecteur pour l'agencement b. Le paramètre n définit le nombre de variables d'optimisation.

L'individu b, parmi les solutions à explorer, qui minimise cette distance d_{a-b} est celui qui correspond à l'individu a. Ce mécanisme de correspondance est illustré sur la figure 4.3 avec un exemple simple à deux variables de conception. Ces variables continues définissent par exemple la position du centre géométrique d'un composant dans le contenant. Elles peuvent varier ici entre les valeurs 10 et 100. Les solutions à explorer sont représentées par des carrés bleus sur la figure 4.3. Supposons un ensemble d'individus appartenant à l'espace de conception. Ces individus sont représentés par des points noirs sur la figure 4.3. Le mécanisme de correspondance est alors illustré par des flèches rouges sur la figure 4.3. La norme utilisée pour le calcul des distances entre individus est ici la norme euclidienne.

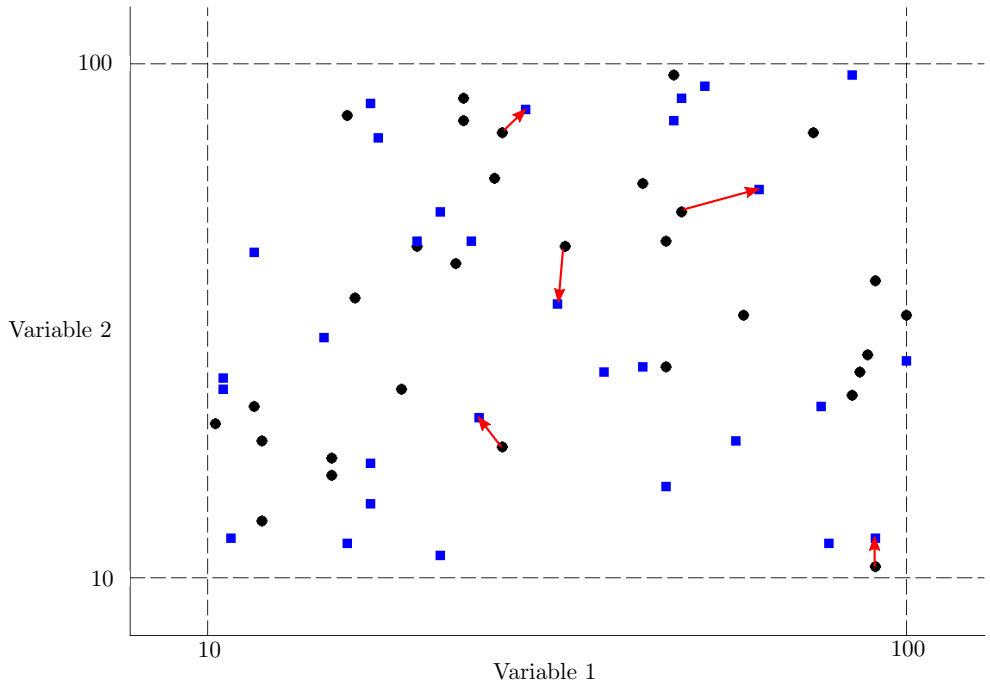


FIGURE 4.3 – *Algorithme Génétique Interactif : espace à explorer vs espace de conception*

4.2.3.3 Principe de fonctionnement de l'exploration perceptive

L'exploration perceptive de solutions est basée sur l'utilisation d'un algorithme génétique interactif. La structure de la méthode est donc très proche de celle d'un algorithme génétique. Elle intègre en plus le mécanisme de correspondance entre l'espace de conception de l'IGA et l'espace de solutions à explorer. Ce mécanisme est décrit dans la partie précédente. La structure du processus d'exploration perceptive de solutions est illustré sur la figure 4.4. L'algorithme génétique utilisée dans ce processus interactif est le MOGA-II (Pol03).

Premièrement, une première population d'agencements, choisis aléatoirement, est présentée au concepteur. Le concepteur est invité à choisir, parmi les solutions présentées, k agencements qui correspondent le mieux à ses critères perceptifs. Via les mécanismes propres à l'algorithme génétique (sélection par roulette, mutation, croisement...), une nouvelle population de solutions est générée. En fait, les solutions choisies par le concepteur ont une probabilité plus élevée d'être sélectionnées pour être l'individu « parent » lors des opérations génétiques. La nouvelle population (pop IGA) générée par l'algorithme génétique est ensuite transformée pour qu'elle corresponde à l'espace des solutions à explorer (pop réelle) et est présentée au concepteur. On réitère ce processus jusqu'à ce que le concepteur soit satisfait des solutions qui lui sont présentées.

Une interface graphique interactive a été développée pour permettre au concepteur d'explorer perceptivement les solutions générées par le concepteur. Cette interface permet d'afficher une population de 8 agencements. Le concepteur a la possibilité de modifier, en cours de processus, les paramètres des opérateurs génétiques. Il peut également obtenir certaines informations sur les agencements,

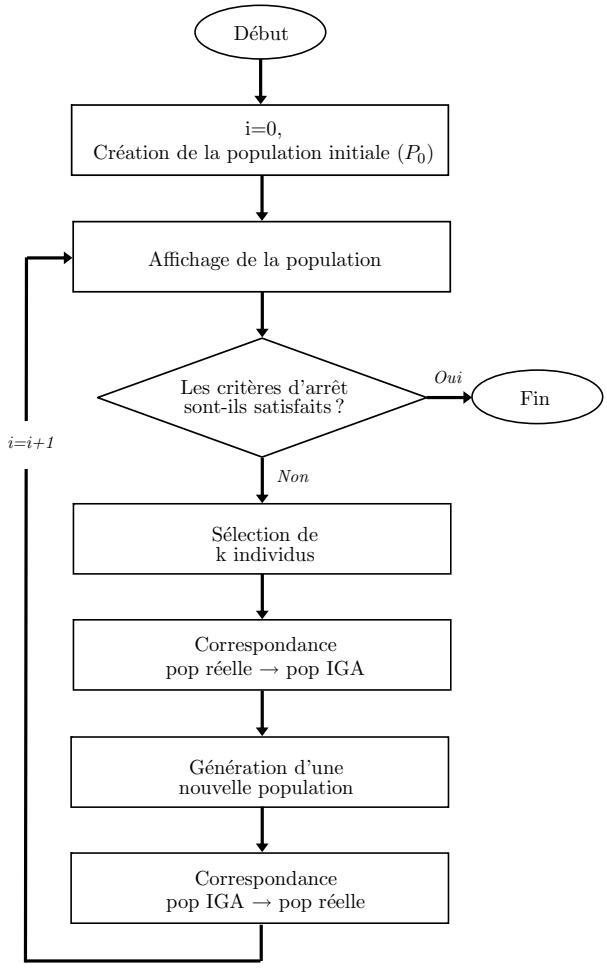


FIGURE 4.4 – *Processus d'exploration perceptive de solutions*

comme par exemple la valeur des contraintes et des objectifs du problème d'optimisation. Cette interface graphique fait également partie du démonstrateur d'optimisation d'agencement, décrit dans le chapitre 5. La figure 4.5 illustre cette interface graphique interactive.

4.3 Interactivité avec une solution

La partie précédente a présenté un certain nombre de méthodes et d'outils interactifs permettant de trier, visualiser, comparer et explorer des solutions d'agencement générées par l'algorithme d'optimisation. Cette première étape consiste donc, pour le concepteur, à choisir une solution répondant à la fois aux exigences du problème et à son jugement personnel. Les deux sous-sections suivantes sont dédiées à l'outil graphique de visualisation et d'interaction avec une solution. Cette interactivité avec le concepteur a pour principal objectif d'améliorer encore les performances d'une solution particulière.

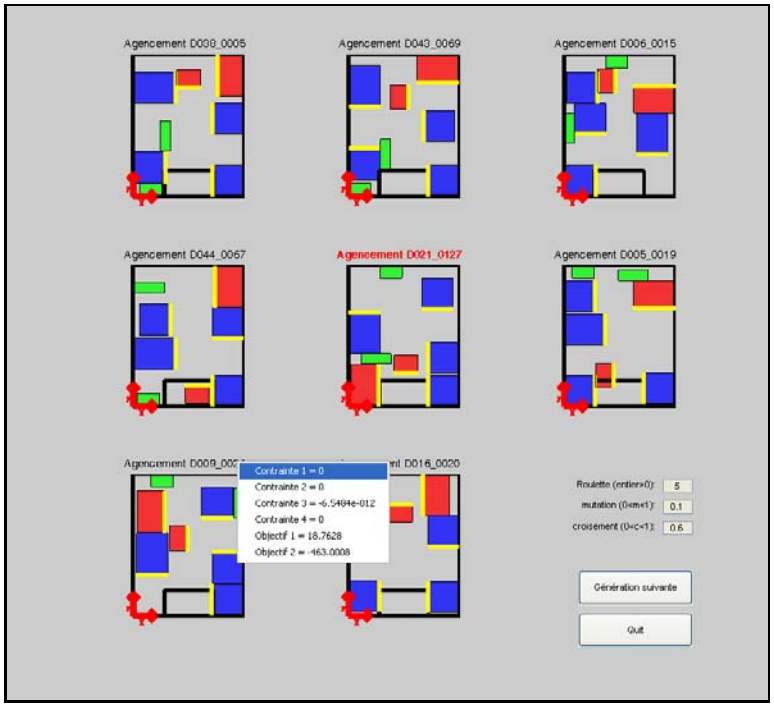


FIGURE 4.5 – *Interface graphique d'exploration perceptive*

4.3.1 Interface interactive et réalité virtuelle

L'interface interactive de visualisation d'une solution a pour objectif de représenter le modèle géométrique de la solution en deux ou trois dimensions et permettre au concepteur d'interagir avec celui-ci. L'interactivité avec le concepteur permet principalement de modifier manuellement et localement le positionnement (position, orientation et sens) d'un ou plusieurs composants de l'agencement. L'objectif est que ces modifications locales apportent, à la solution proposée par l'algorithme, l'expertise et le jugement personnel du concepteur. La figure 4.6 illustre l'interface graphique présente dans le démonstrateur d'optimisation d'agencement. Le problème d'agencement, pris pour exemple, est un problème d'agencement en trois dimensions, qui sera détaillé dans le chapitre 5 relatif aux applications industrielles.

En résumé, les principales fonctionnalités de l'interface graphique interactive sont :
- visualiser le modèle géométrique de la solution initialement proposée par l'algorithme d'optimisation,
- modifier localement la position, l'orientation et le sens des composants,
- exécuter une optimisation locale à partir d'une solution modifiée localement par le concepteur,
- visualiser le modèle géométrique de la solution modifiée par le concepteur,
- avoir un retour instantané sur les valeurs des contraintes et des objectifs du problème d'optimisation et ainsi établir une comparaison objective entre la solution générée par l'algorithme d'optimisation et la solution modifiée par le concepteur.

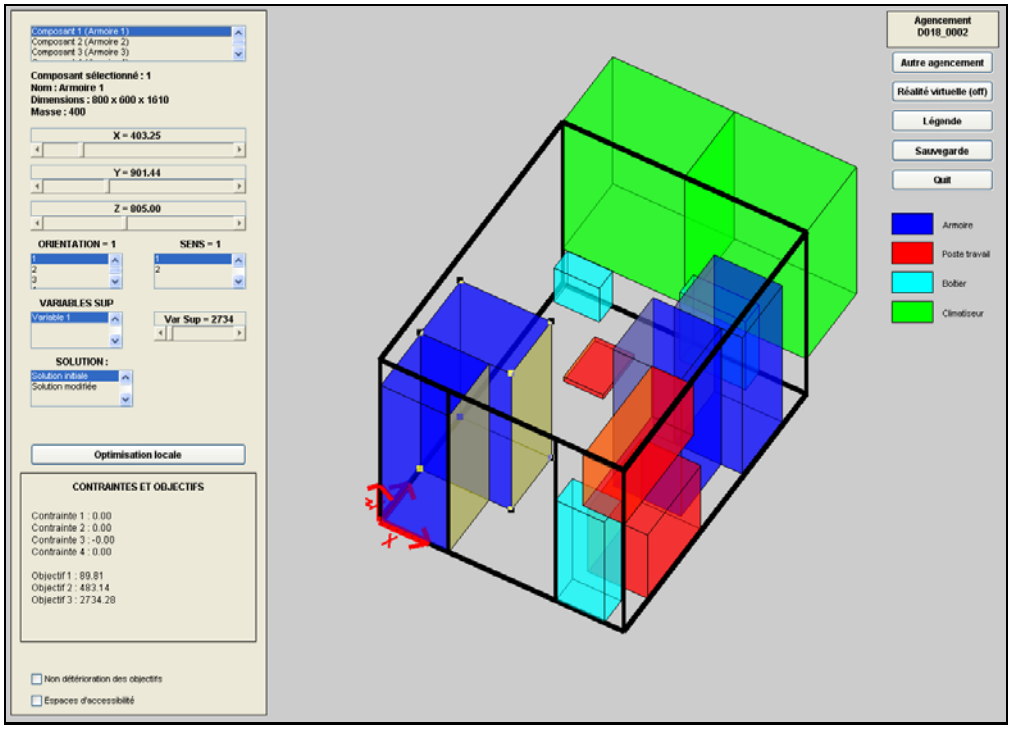


FIGURE 4.6 – *Interface interactive de visualisation et modification d'une solution*

L'optimisation locale consiste à résoudre un problème d'optimisation mono-objectif sous contraintes. Les contraintes correspondent aux contraintes de conception du problème d'optimisation d'agencement. Le critère d'optimisation est défini par la somme pondérée (coefficients de pondération fixés arbitrairement par le concepteur) des différents objectifs d'optimisation du problème d'agencement. Les variables d'optimisation sont les variables continues qui définissent le positionnement des composants à l'intérieur du contenant.

D'autres fonctionnalités sont également proposées dans cette interface interactive. Il est notamment possible de visualiser, s'ils existent, les espaces d'accessibilité des composants, de modifier les couleurs des composants, de modifier la légende du modèle géométrique ou encore de sauvegarder la solution modifiée par le concepteur.

Un environnement de réalité virtuelle a également été implémenté dans l'interface graphique de visualisation et de modification d'une solution. En effet, pour certains problèmes d'agencement, notamment pour les applications en trois dimensions, il est parfois difficile de manipuler certains composants. Ceux-ci peuvent être positionnés derrière d'autres composants et il est alors impossible d'apprécier les éventuelles collisions avec le contenant ou les autres composants. Le seul indicateur disponible est la valeur des contraintes de chevauchement qui peut varier en fonction de la position de ce composant. Il a donc été associé, à l'interface interactive de visualisation d'une solution, une interface haptique du type joystick à retour d'efforts. Le joystick utilisé est semblable à celui représenté sur la figure 4.7.

FIGURE 4.7 – *Joystick à retour d'efforts*

Il est donc possible, via ce joystick à retour d'efforts, de déplacer le composant et modifier son orientation et son sens. Le retour d'efforts permet au concepteur d'apprécier les collisions du composant avec les autres composants et le contenant. L'immersion du concepteur dans son problème d'agencement est alors améliorée.

4.3.2 Indicateur d'aide à la modification d'une solution

L'interface graphique, présentée précédemment, comporte les principaux outils d'interaction entre le concepteur et une solution d'agencement. Le concepteur peut donc modifier localement une solution et évaluer directement les conséquences de ses modifications sur les contraintes et les objectifs du problème d'optimisation. L'objectif de l'indicateur, proposé dans cette partie, est de permettre au concepteur de savoir à priori comment modifier de manière efficace la solution proposée par l'algorithme. Il ne sait pas comment déplacer un composant de manière à ne pas détériorer la solution, en termes de contraintes et d'objectifs, tout en répondant au mieux à sa propre expertise et son jugement personnel sur la solution.

L'idée principale, décrite dans ce paragraphe, est donc de proposer au concepteur un indicateur d'aide à la modification d'une solution. L'objectif de cet indicateur est de représenter les zones, situées autour d'un composant sélectionné par le concepteur, dans lesquelles le composant peut être déplacé sans détériorer la solution d'agencement. Une solution modifiée est détériorée par rapport à une solution courante si une des conditions suivantes est vérifiée :
– les contraintes de la solution modifiée sont moins bien respectées que pour la solution courante,
– la solution modifiée est dominée, au sens de la Pareto-dominance, par la solution courante.

La figure 4.8 illustre le principe de l'indicateur d'aide à la modification d'une solution, pour un problème en deux dimensions où le composant est représenté par un rectangle bleu. Deux zones différentes sont représentées : une zone blanche définissant la zone de non-détérioration de la solution

courante et une zone verte indiquant la zone d'amélioration de la solution courante. En effet, si le composant se déplace dans la zone verte, zone d'amélioration, les contraintes de la solution modifiée sont aussi bien voire mieux respectées que pour la solution courante et la solution modifiée domine, au sens de la Pareto dominance, la solution courante. Comme l'indique la figure 4.8, l'algorithme de construction de ces deux zones est basé sur l'exploration de l'espace autour du composant, suivant deux directions : la direction radiale paramétrée par R et la direction angulaire paramétrée par θ. Pour chaque couple (R, θ), l'algorithme teste si la nouvelle position du composant engendre une détérioration ou une amélioration de la solution.

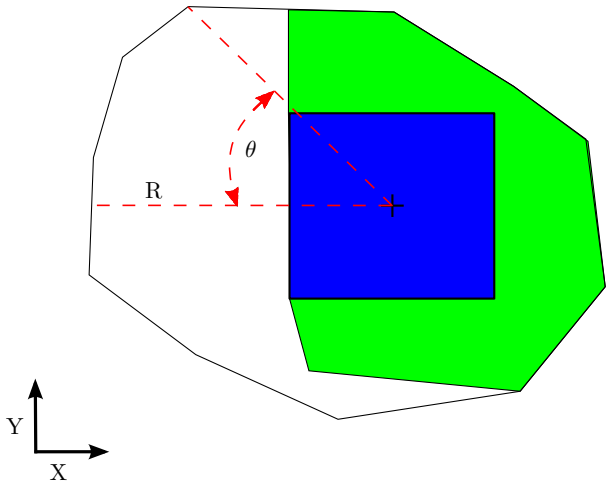


FIGURE 4.8 – *Indicateur 2D d'aide à la modification d'une solution*

Aussi, il est possible d'exporter le principe de construction de ces zones de modification d'une solution, aux problèmes d'agencement en trois dimensions. Pour cela, on ajoute à l'algorithme de construction de ces zones, une boucle supplémentaire permettant de déplacer le composant sur l'axe vertical $(O, \overrightarrow{Z})$. L'idée est donc de déplacer le composant dans un cylindre, centré sur la position courante du composant, et d'évaluer les nouvelles performances de l'agencement, issues du déplacement du composant.

4.4 Conclusion

Ce chapitre 4 a présenté un ensemble de méthodes et d'outils qui permettent au concepteur de faire un choix parmi l'ensemble des alternatives de conception proposées par l'algorithme d'optimisation. Tout d'abord, la stratégie proposée dans ce chapitre consiste à trier toutes les solutions générées par l'algorithme : solutions admissibles, variantes ou encore solutions Pareto-optimales. Ensuite, le choix du concepteur repose sur une visualisation multidimensionnelle des solutions. Cette visualisation permet de comparer les solutions en analysant les informations quantitatives qui les caractérisent (contraintes, objectifs, variables, modèle géométrique, rang). Deux outils graphiques, basés sur ce concept de « *design by shopping* » sont décrits : le nuage de points et le graphe parallèle.

De plus, ce chapitre présente une stratégie innovante d'exploration de solutions d'agencement. Cette exploration est perceptive, c'est-à-dire qu'elle prend en compte les critères subjectifs et perceptifs du concepteur. Ces critères correspondent généralement à des exigences qui n'ont pas pu être intégrées dans la formulation du problème d'optimisation d'agencement. L'exploration perceptive de ces solutions est basée sur l'utilisation d'un algorithme génétique interactif (IGA) qui associe les choix du concepteur à la génération des solutions.

Aussi, lorsque le choix du concepteur s'est arrêté sur une solution particulière, la démarche présentée dans ce chapitre suggère au concepteur d'interagir avec cette solution. Cette interactivité a pour objectif de modifier manuellement et localement le positionnement de certains composants afin d'améliorer les performances de la solution choisie. Ces modifications, faites par le concepteur, sont facilitées par l'utilisation d'un indicateur d'aide à la modification de la solution. Pour un composant préalablement sélectionné par le concepteur, cet indicateur représente les zones dans lesquelles le composant peut être déplacé sans détériorer le solution courante. Enfin, pour les problèmes d'optimisation d'agencement en trois dimensions, un joystick à retour d'efforts a été implémenté dans cette interface graphique interactive. Cet outil de réalité virtuelle permet alors au concepteur d'être davantage immergé dans la conception de l'agencement.

5 Applications industrielles

5.1 Introduction

Ce chapitre est dédié aux applications industrielles de la démarche d'optimisation d'agencement proposée dans ce manuscrit. Pour chaque cas d'étude présenté dans ce chapitre, toutes les étapes de la méthode sont décrites : la description, la formulation, la résolution du problème, ainsi que les choix de conception faits par le concepteur.

Les deux premiers problèmes d'optimisation d'agencement présentés dans ce chapitre portent sur un exemple industriel proposé par l'entreprise française Thales Communications & Security, implantée à Cholet. Ce cas d'application porte sur l'agencement d'un shelter. Un shelter est un abri technique mobile dans lequel sont disposés des équipements, tels des armoires, des bureaux et autres boîtiers électriques. Ce local technique est le plus souvent fixé à l'arrière d'un véhicule et est dédié à des missions de communications lors d'opérations militaires. Le premier exemple traite le problème d'agencement du shelter en deux dimensions et fait l'objet de plusieurs études (BBPR10b; BBPR10c). Le second cas d'étude est une extension du premier problème d'agencement en trois dimensions.

Le troisième problème d'optimisation d'agencement proposé dans ce chapitre porte sur un problème en trois dimensions de stockage de composants dans un contenant. Ce cas d'application, par rapport aux deux premiers problèmes, met en avant l'utilisation du module d'optimisation GRAV, qui prend en compte la gravité pour les problèmes d'agencement en trois dimensions. Ce module est décrit dans le chapitre 3. Enfin, la dernière partie de ce chapitre est décrit brièvement le démonstrateur d'optimisation d'agencement qui a été développé au cours de ce doctorat.

5.2 Agencement d'un shelter en 2D

Ce cas d'étude porte sur l'agencement d'un shelter. Huit équipements sont à disposer dans ce shelter : quatre armoires, deux bureaux et deux boîtiers électriques. Une vue CAO du shelter en trois dimensions est illustrée sur la figure 5.1. Les dimensions du shelter et des équipements, ainsi que la masse de chaque équipement, sont indiquées dans le tableau 5.1. Les parties suivantes décrivent les différentes étapes de la méthode, proposée dans ce manuscrit, qui ont permis de résoudre ce problème d'agencement.

N°	Contenant/Composant	Dim (mm)	Masse (kg)
0	Shelter	2150×2740×1910	
1	Armoire 1	800×600×1610	400
2	Armoire 2	600×600×1610	300
3	Armoire 3	600×600×1610	300
4	Armoire 4	600×600×1610	300
5	Bureau 1	350×465×50	10
6	Bureau 2	525×800×750	30
7	Boîtier 1	200×580×800	45
8	Boîtier 2	250×430×1185	35

TABLE 5.1 – *Shelter 2D : dimensions et masse du shelter et des équipements*

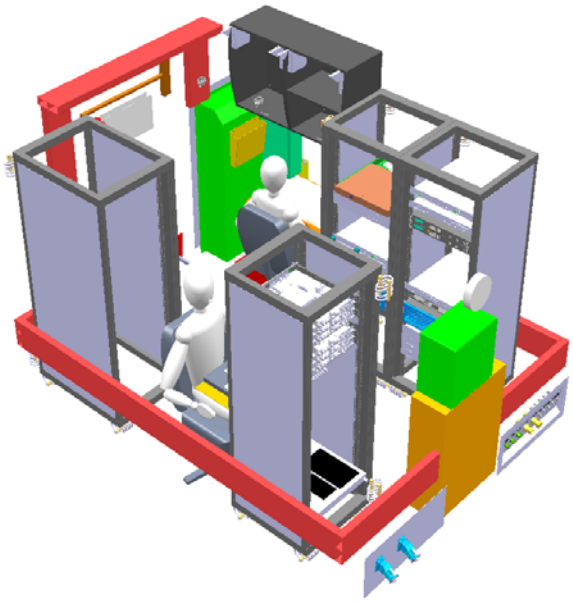


FIGURE 5.1 – *Vue CAO 3D du shelter*

5.2.1 Description du problème

Ce problème d'agencement est un problème de placement en trois dimensions d'équipements dans un local. Le problème a tout d'abord été simplifié en deux dimensions pour deux raisons :
- la hauteur des armoires correspond approximativement à la hauteur du shelter et ne permet pas la superposition des boîtiers sur les armoires,
- il n'est pas possible de placer les boîtiers électriques au-dessus des bureaux.

Le modèle géométrique en deux dimensions du shelter est illustré sur la figure 5.2. Comme le suggère le chapitre 2, les composants peuvent être séparés en deux catégories :
- composants matériels : 4 armoires, 2 bureaux et 2 boîtiers électriques
- composants virtuels :
  - 6 espaces d'accessibilité des armoires et des bureaux (représentés en traits pointillés sur la figure 5.2),
  - 1 espace libre devant la porte (non représenté sur la figure 5.2),
  - 1 espace libre sous l'entrée du climatiseur extérieur (représenté par une zone hachurée sur la figure 5.2) dans lequel aucune armoire ne peut être disposée,
  - 1 couloir central (représenté par un trait mixte sur la figure 5.2), garantissant l'accès aux équipements du shelter.

Comme expliqué dans le chapitre 2, les espaces d'accessibilité des armoires sont modélisés pour garantir un espace libre devant les armoires permettant d'introduire du matériel dans celles-ci. Les espaces d'accessibilité des bureaux correspondent aux espaces occupés par l'opérateur devant son poste de travail. Enfin, on considère dans ce premier modèle que les dimensions du contenant et des

composants restent fixes durant l'optimisation d'agencement. Finalement, ce problème d'agencement comporte 17 composants (8 éléments matériels et 9 éléments virtuels).

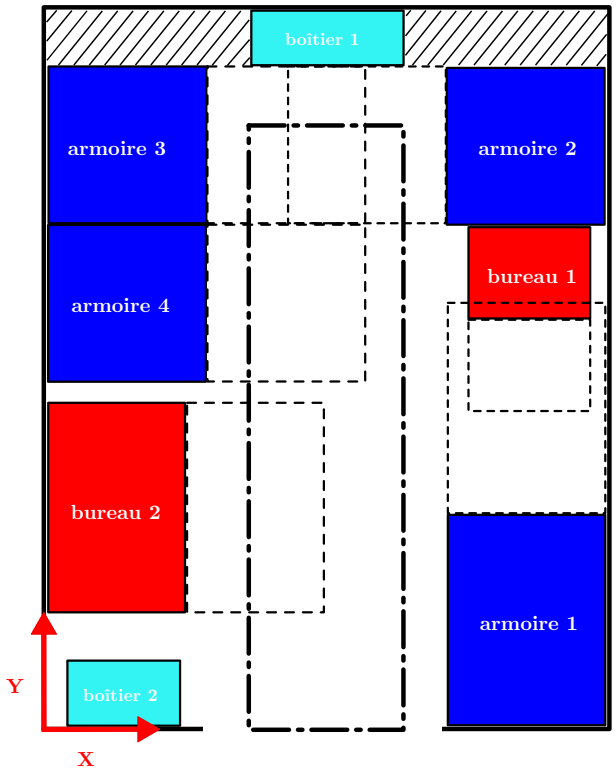


FIGURE 5.2 – *Modèle géométrique 2D du problème d'agencement du shelter*

5.2.2 Formulation du problème

L'objectif de cette étape est de traduire le modèle géométrique décrit précédemment et les différentes exigences du concepteur afin de formuler un problème d'optimisation d'agencement, c'est-à-dire définir des variables de conception, des contraintes et des critères d'optimisation.

5.2.2.1 Variables de conception

Les variables de conception définissent la position, l'orientation et le sens de chaque composant. Selon les exigences du concepteur, ces variables sont définies de la manière suivante :

- 2 variables continues (X et Y) et 2 variables discrètes (α et λ) pour la position, l'orientation et le sens de chaque armoire (4) et de chaque bureau (2),
- 2 variables continues (X et Y) et 1 variable discrète (α) pour la position et l'orientation du boîtier 1,
- 1 variable continue (X) et 1 variable discrète (Y) pour la position du boîtier 2.

Les variables X et Y varient respectivement de 0 à 2150 mm et de 0 à 2740 mm, correspondant aux dimensions du shelter. Pour chaque composant, il est possible de réduire ces intervalles de définition en fonction de la plus petite des dimensions du composant. Étant donnée la forme rectangulaire des composants, α peut prendre deux valeurs 0 et 90° (ou 1 et 6 selon la représentation adoptée dans cette approche et illustrée sur la figure 2.3), correspondant aux deux orientations possibles des composants en deux dimensions. La variable λ est seulement définie pour les composants qui possèdent un espace d'accessibilité. Étant donnée une orientation α du composant, il y a deux sens λ possibles : 1 ou 2.

Le positionnement du boîtier 2 est soumis à des exigences particulières. En effet, le concepteur souhaite que celui-ci soit nécessairement fixé sur la paroi avant ou arrière du shelter. La coordonnée Y du centre géométrique du boîtier 2 peut prendre donc deux valeurs possibles (100 ou 2615 mm) et la variable X est une variable continue, définissant la position du boîtier 2 sur une de ces deux parois. En résumé, ce problème d'optimisation comporte 29 variables combinant des variables continues et discrètes.

5.2.2.2 Contraintes de conception

Les contraintes définies par le concepteur sont uniquement des contraintes de non-chevauchement entre composants. Ces contraintes sont divisées en quatre catégories :
- C1 : non-chevauchement entre les composants matériels,
- C2 : non-chevauchement entre les composants matériels et les espaces d'accessibilité,
- C3 : non-chevauchement entre tous les composants (matériels et virtuels) et l'extérieur du shelter, l'espace devant la porte et le couloir central,
- C4 : non-chevauchement entre les armoires et l'espace sous l'entrée du climatiseur.

Il n'existe pas de contrainte de non-chevauchement entre les espaces d'accessibilité car on suppose que les opérations de chargement de matériel dans les armoires sont séquentielles et que deux personnes ne travaillent jamais simultanément dans le shelter. Deux espaces d'accessibilité peuvent donc se recouvrir. On suppose également que les espaces d'accessibilité peuvent chevaucher les différents espaces libres présents dans le shelter. Aussi, il existe une particularité pour le bureau 1. Ce bureau est une tablette qui peut se rabattre et on suppose donc qu'elle peut chevaucher tous les composants virtuels présents dans le shelter. Étant donnée la forme rectangulaire des composants, les contraintes de non-chevauchement sont évaluées via le calcul de l'aire d'intersection entre les composants. L'aire d'intersection entre les composants i et j est donnée par la formule 2.2 (chapitre 2).

5.2.2.3 Objectifs d'optimisation

Les objectifs de ce problème d'optimisation traduisent les exigences fonctionnelles formulées par le concepteur. Ces exigences sont :

– équilibrer les masses à l'intérieur du shelter,
– éloigner le réseau « énergie » (symbolisé par l'armoire 1) du réseau « signaux » (symbolisé par les armoires 3 et 4 et le boîtier 2).

Deux objectifs d'optimisation ont donc été définis pour traduire ces exigences :
– **minimiser** (F1), avec F1 : distance entre le centre de gravité de l'ensemble des composants et le centre géométrique du shelter,
– **maximiser** (F2), avec F2 : distance entre l'armoire 1 et l'ensemble composé des armoires 3 et 4 et du boîtier 2.

Toutes les distances calculées dans l'évaluation de ces objectifs sont définies par la norme euclidienne. La distance entre les composants i et j est donc formulée par :

$$d_{ij} = \sqrt{(x_j - x_i)^2 + (y_j - y_i)^2} \tag{5.1}$$

où (x_i, y_i) définissent les coordonnées du centre géométrique du composant i. Lorsqu'il est nécessaire de calculer une distance par rapport à un ensemble de composants, la somme des distances à chaque composant est évaluée.

Le calcul des coordonnées du centre de gravité de l'ensemble des composants est donnée par les formules suivantes :

$$X_{gra} = \frac{\sum_{i=1}^{N}(x_i \times m_i)}{\sum_{i=1}^{N} m_i} \tag{5.2}$$

$$Y_{gra} = \frac{\sum_{i=1}^{N}(y_i \times m_i)}{\sum_{i=1}^{N} m_i} \tag{5.3}$$

où N représente le nombre de composants qui entrent dans le calcul de ce centre de gravité. Dans les équations 5.2 et 5.3, le paramètre m_i représente la masse du composant i.

5.2.3 Indice de faisabilité du problème d'agencement

Nous supposons tout d'abord que les espaces d'accessibilité des composants matériels ont des dimensions identiques à ces mêmes composants matériels. Les dimensions des autres composants virtuels, correspondant aux espaces libres, sont indiquées dans le tableau 5.2.

Composant virtuel (espace libre)	Dim (mm)
Espace devant la porte	900×500×1910
Espace sous l'entrée du climatiseur	2150×200×1910
Couloir central	500×2100×1910

TABLE 5.2 – *Shelter 2D : dimensions des espaces libres*

La formulation de l'indice de faisabilité du problème d'agencement est présentée dans le chapitre 2. En tenant compte des contraintes formulées par le concepteur, la matrice d'intersections entre les différents composants de l'agencement est égale à :

$$M_{inter} = \begin{vmatrix} 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ & & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ & & & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ & & & & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ & & & & & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 \\ & & & & & & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 \\ & & & & & & & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 \\ & & & & & & & & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ & & & & & & & & & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ & & & & & & & & & & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ & & & & & & & & & & & 0 & 0 & 0 & 0 & 0 & 0 \\ & & & & & & & & & & & & 0 & 0 & 0 & 0 & 0 \\ & & & & & & & & & & & & & 0 & 0 & 0 & 0 \\ & & & & & & & & & & & & & & 0 & 0 & 0 \\ & & & & & & & & & & & & & & & 0 & 0 \\ & & & & & & & & & & & & & & & & 0 \end{vmatrix} \quad (5.4)$$

Cette matrice d'intersections étant symétrique, la partie triangulaire supérieure est uniquement indiquée dans l'équation 5.4. Aussi, dans cette matrice d'intersections, les composants de l'agencement sont classés de la première ligne à la dernière ligne dans cet ordre : composants matériels (8), espaces d'accessibilité (6), espace libre devant la porte (1), espace libre sous l'entrée du climatiseur (1) et couloir central (1), soit 17 composants. Un exemple de lecture de cette matrice d'intersections est le suivant : la 6ème ligne (ou 6èmecolonne) de la matrice signifie que le composant n°6 (bureau 2) ne peut pas chevaucher les autres composants matériels, les espaces d'accessibilité, l'espace libre devant la porte et le couloir central mais qu'il peut chevaucher l'espace sous l'entrée du climatiseur.

En appliquant la méthode de calcul exposée dans le chapitre 2, nous obtenant un indice de faisabilité égal à 66,26 %, ce qui montre à priori la faisabilité du problème d'optimisation d'agencement. Par ailleurs, si les composants de cet agencement avaient tous été considérés comme des composants matériels et si nous avions appliqué la formulation traditionnelle de la compacité, la compacité du problème serait égale à 105,6 %, démontrant à priori l'infaisabilité du problème d'agencement.

5.2.4 Résolution du problème et comparaison des stratégies d'optimisation

5.2.4.1 Utilisation de l'algorithme génétique seul

Dans un premier temps, l'algorithme génétique seul est testé sur ce problème d'optimisation d'agencement du shelter avec un couloir central. L'algorithme génétique ici utilisé est l'Omni-Optimizer proposé par Deb *et al.* (DT08). Cet algorithme a été conçu pour résoudre des problèmes d'optimisation mono-objectif et multiobjectif. La structure de cet algorithme est basée sur celle du NSGA-II (DPAM02). Sa particularité réside dans la présence de nouveaux mécanismes qui lui permettent de mieux converger dans certains cas. Par exemple, l'algorithme Omni-Optimizer prend en compte les distances phénotypiques (espace des objectifs) et génotypiques (espace des variables), lui permettant de conserver une bonne diversité des solutions.

L'algorithme Omni-Optimizer est donc initialisé par 200 individus tous aléatoirement générés. Les coefficients des opérateurs de croisement et de mutation pour les variables continues et discrètes sont respectivement fixés à 0,7 et 0,1. Le nombre d'itérations maximal autorisé pour la génération d'individus est fixé à 100. On réitère dix fois le processus de l'algorithme génétique Omni-Optimizer pour prendre en compte le comportement stochastique de l'algorithme. Le graphique illustré sur la figure 5.3 représente la convergence moyenne de cet algorithme sur ce problème d'optimisation d'agencement du shelter.

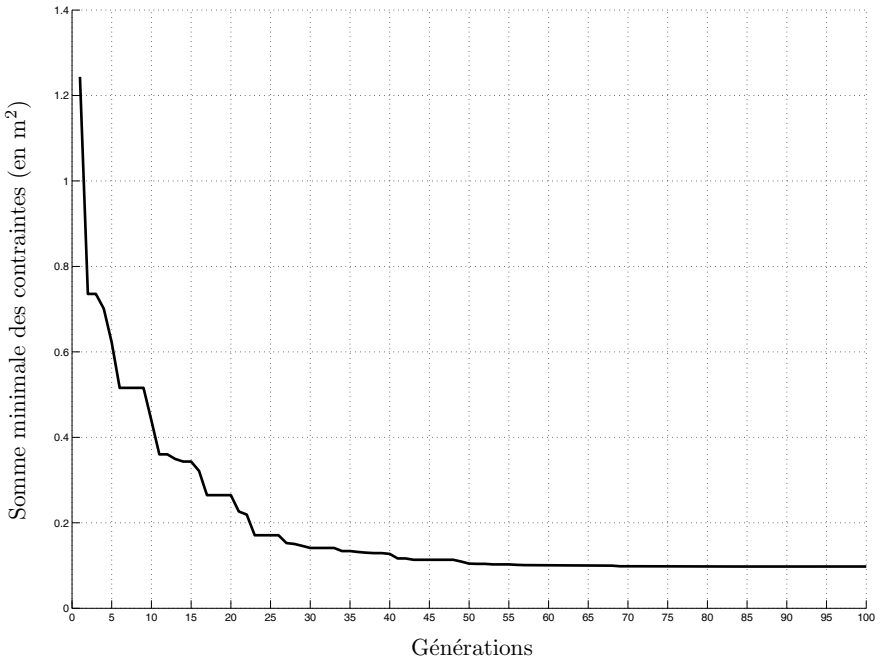


FIGURE 5.3 – *Convergence de l'algorithme génétique Omni-Optimizer sur le problème d'agencement du shelter 2D avec couloir*

Afin de réaliser ce graphe de convergence, on calcule tout d'abord la somme des contraintes de conception (aire de chevauchement en m^2) pour tous les individus générés par l'algorithme. On représente ensuite sur la figure 5.3, le somme minimale obtenue par un individu pour chaque génération de l'algorithme. Si cette somme minimale est égale à 0 pour une génération donnée, cela signifie qu'il existe au moins un individu, présent dans cette génération, qui est une solution admissible du problème d'optimisation d'agencement d'espace. En effet, on considère ici que le seuil de respect des contraintes, défini dans le chapitre 4, est fixé à $\triangle c = 0\,m^2$ pour chaque contrainte de non-chevauchement du problème.

Sur la figure 5.3, nous pouvons donc remarquer que l'algorithme génétique Omni-Optimizer n'a pas trouvé de solution admissible pour ce problème d'optimisation d'agencement. Il existe des individus qui respectent les contraintes C1, C2 et C4 mais la contrainte C3 n'est respectée par aucun individu. Ce problème est donc trop complexe pour l'algorithme génétique Omni-Optimizer. Il est donc nécessaire d'associer à cet algorithme des modules d'optimisation supplémentaires qui vont l'aider dans sa recherche de solutions optimales.

5.2.4.2 Comparaison des algorithmes issus de la stratégie modulaire

Cette partie vise à comparer les performances des différents algorithmes qui sont issus de la stratégie modulaire, c'est-à-dire de l'association de l'algorithme génétique avec différents modules d'optimisation. Ces modules d'optimisation ont été définis dans le chapitre 3 et sont rappelés ici :
- module OPEA : optimisation du placement des espaces d'accessibilité,
- module SEPA : algorithme de séparation,
- module PERT : perturbation locale,
- module GRAV : prise en compte de la gravité.

Le module GRAV n'est ici pas utilisé car le problème d'agencement traité est en deux dimensions. Plusieurs algorithmes résultent donc de la combinaison de l'algorithme génétique avec un ou plusieurs de ces modules d'optimisation. C'est le principe de la stratégie modulaire proposée dans le chapitre 3. Nous considérons ici quatre algorithmes différents résultant de cette stratégie modulaire :
- algorithme A : algorithme génétique + module OPEA
- algorithme B : algorithme génétique + module SEPA
- algorithme C : algorithme génétique + module OPEA + module SEPA
- algorithme D : algorithme génétique + module OPEA + module SEPA + module PERT (avec $j_{max} = 3$)

Le paramètre j_{max} est utilisé par le module d'optimisation PERT et représente ici le nombre maximal de tentatives de recherche d'une solution admissible. Le seuil d'activation des modules SEPA et PERT, correspondant à un seuil de respect des contraintes de placement, est fixé à $10\,cm^2$. Ceci signifie que si pour un agencement donné, la valeur maximale des contraintes de placement est supérieure à $10\,cm^2$, les modules SEPA et PERT sont activés. Le nombre maximal d'itérations de

l'algorithme d'optimisation utilisé dans le module SEPA est fixé à 100. Aussi, pour le module PERT, le nombre de composants dont l'orientation ou le sens change, est fixé à 3.

Afin de comparer les performances de ces quatre algorithmes, ces derniers sont initialisés par la même population initiale que l'algorithme génétique. Les paramètres génétiques de chaque algorithme sont identiques à ceux de l'algorithme génétique seul. On réitère également dix fois le processus de chaque algorithme pour prendre en compte le comportement stochastique des algorithmes. Le seul paramètre qui évolue entre les algorithmes d'optimisation est le nombre maximal d'itérations autorisé. En effet, parce que chaque algorithme n'évolue pas à la même vitesse et afin d'homogénéiser les résultats sur l'échelle du temps de calcul, le nombre maximal d'itérations pour l'algorithme A est fixé à 300, pour les algorithmes B et C à 200 et pour l'algorithme D à 100.

La comparaison des ces algorithmes est effectuée suivant deux aspects : la quantité de variantes admissibles générées et la qualité des solutions trouvées par rapport au problème d'optimisation posé.

Comparaison quantitative

Une première comparaison de ces algorithmes est réalisée en prenant en compte le nombre de variantes admissibles générées au cours du temps. La figure 5.4 illustre cette première comparaison.

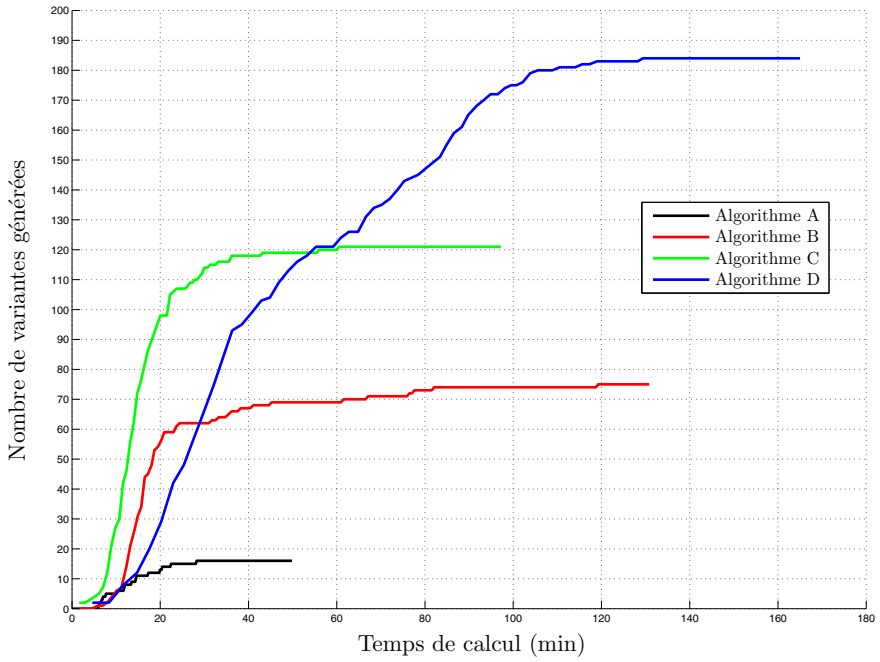


FIGURE 5.4 – *Problème d'agencement du shelter 2D avec couloir : comparaison quantitative des performances des algorithmes d'optimisation*

La figure 5.4 illustre donc l'évolution du nombre de variantes admissibles générées par chaque

algorithme, en fonction du temps de calcul. La notion de variante est définie dans le chapitre 4. Afin de détecter ces variantes, les paramètres $\triangle g$ (différence géométrique) et $\triangle f$ (différence fonctionnelle) sont respectivement fixés à $50\,cm$ et $10\,cm$. Cette figure met en évidence l'influence des modules d'optimisation sur la recherche de solutions optimales. En effet, la figure 5.4 montre tout d'abord que tous les algorithmes d'optimisation ont trouvé au moins une variante admissible pour ce problème d'optimisation. Par exemple, l'algorithme A, qui combine l'algorithme génétique avec le module d'optimisation OPEA a trouvé 16 variantes admissibles après 300 générations. Ce nombre de variantes augmente en fonction de l'algorithme utilisé. En résumé, les algorithmes peuvent être classés suivant le nombre total de variantes générées dans l'ordre croissant suivant : algorithmes A, B, C et D.

Il est également possible d'affirmer, via les résultats présentés sur la figure 5.4, que l'algorithme C a de meilleurs performances que l'algorithme B qui a lui-même de meilleurs performances que l'algorithme A, car tout au long du processus d'optimisation, le nombre de variantes générées par l'algorithme C est supérieur à celui de l'algorithme B qui est lui même supérieur à celui de l'algorithme A. Pour l'algorithme D, il est plus difficile de faire un choix. Ce choix dépend fortement des exigences du concepteur en terme d'obtention de résultats. En effet, l'algorithme D est celui qui génère le plus de variantes, mais c'est également celui qui dépense le plus de temps de calcul dans la génération de ces variantes. En effet, on peut dire par exemple que l'algorithme C a de meilleures performances que l'algorithme D si le temps de calcul est inférieur à environ 55 minutes et inversement ensuite. Les modules d'optimisation améliorent nettement la recherche de solutions mais le temps de calcul de l'algorithme est proportionnel au taux d'utilisation de ces modules d'optimisation.

Aussi, il est intéressant d'évaluer l'influence du paramètre j_{max} pour l'algorithme résultant de la combinaison de l'algorithme génétique avec les modules d'optimisation OPEA, SEPA et PERT (algorithme D). La figure 5.5 montre l'évolution du nombre de variantes générées par l'algorithme D en fonction du temps de calcul, pour différentes valeurs du paramètre j_{max} (3 et 10).

Sur la figure 5.5, on peut remarquer que l'algorithme D avec le paramètre j_{max} réglé à 10 génère un plus grand nombre de variantes admissibles (207 variantes) alors que le même algorithme avec $j_{max} = 3$ génère 184 variantes (soit 12,5 % de variantes en plus). Cependant, parce que les modules SEPA et PERT requièrent du temps de calcul dans la recherche de solutions faisables, l'algorithme D avec $j_{max} = 10$ nécessite plus de temps de calcul pour générer les variantes. Finalement, le choix du concepteur sur l'algorithme d'optimisation le plus adapté à son problème et à ses propres exigences résulte d'un compromis entre le nombre d'alternatives de conception trouvées et le temps de calcul nécessaire à la création de ces alternatives.

Comparaison qualitative

Une seconde étape consiste à comparer ces quatre algorithmes selon la qualité des solutions optimales obtenues. Pour cela, on se propose d'utiliser deux des indicateurs de comparaison des surfaces de compromis d'un problème d'optimisation : l'hypervolume relatif (HVR) et la métrique $\mathcal{C}$. Ces deux

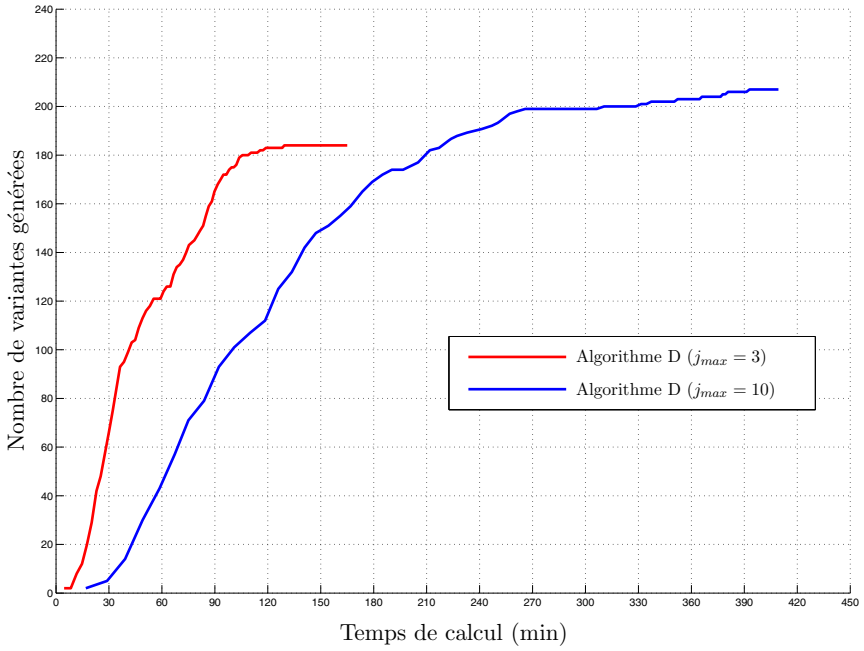


FIGURE 5.5 – *Comparaison des performances de l'algorithme D suivant la valeur de* j_{max}

indicateurs sont décrits dans le chapitre 1, relatif à l'état de l'art.

Afin de comparer la qualité des solutions optimales obtenues pour chaque algorithme, l'hypervolume de chaque ensemble de solutions est normé à partir de l'hypervolume de l'ensemble des solutions optimales générées par l'ensemble des algorithmes testés. Pour cela, les solutions admissibles Pareto-optimales sont calculées pour chaque algorithme. Le tableau 5.3 indique pour chaque algorithme testé le nombre de solutions admissibles et le nombre de solutions admissibles Pareto-optimales. Parce que le nombre d'individus générés n'est pas le même pour chaque algorithme (car le nombre de générations n'est pas le même), nous indiquons également le pourcentage correspondant de solutions par rapport à la population de départ. Pour les solutions admissibles, la population de départ est la population totale générée par l'algorithme d'optimisation et pour les solutions admissibles Pareto-optimales, la population de départ est la population des solutions admissibles.

Algorithmes	Solutions admissibles	Solutions admissibles Pareto-optimales
A	51384 (85,64 %)	1049 (2,04 %)
B	37587 (93,97 %)	279 (0,74 %)
C	37854 (94,64 %)	336 (0,89 %)
D	18132 (91,56 %)	93 (0,51 %)

TABLE 5.3 – *Solutions admissibles et Pareto-optimales pour le shelter 2D avec couloir*

L'ensemble des solutions admissibles Pareto-optimales obtenues par tous les algorithmes testés est

donc composé de 1757 solutions. Parmi ces 1757 solutions, 343 solutions sont Pareto-optimales, soit 19,52 % des solutions. En prenant comme point de référence $W = (10^4, 0)$, l'hypervolume correspondant à ces 343 solutions admissibles Pareto-optimales est égal à $6,78 \times 10^4$.

Ensuite, en considérant le même point de référence, l'hypervolume des solutions optimales trouvées par chaque algorithme est calculé. Ces hypervolumes sont ensuite normés par rapport à l'hypervolume de référence, égal à $6,78 \times 10^4$. Ces hypervolumes relatifs (HVR) sont regroupés dans le tableau 5.4.

Algorithmes	HVR
A	66,86 %
B	100 %
C	98,11 %
D	98,12 %

TABLE 5.4 – *Hypervolumes normés pour le problème du shelter 2D avec couloir*

Le tableau 5.4 montre que la surface de compromis (solutions admissibles Pareto-optimales) est de meilleur qualité pour l'algorithme B. Néanmoins, les valeurs rapprochées pour les hypervolumes relatifs correspondant aux algorithmes C et D indiquent des surfaces de compromis de qualité très proche. Le tableau 5.4 indique également que les solutions admissibles Pareto-optimales trouvées par l'algorithme A ont des performances moindres.

Aussi, il est possible de comparer qualitativement les surfaces de compromis des algorithmes en utilisant la métrique $\mathcal{C}$, décrite dans le chapitre 1. Cet indicateur, calculé entre deux surfaces de compromis A et B, évalue la proportion de solutions de B qui sont dominées par les solutions de A. Le tableau 5.5 indique les différentes valeurs des métriques $\mathcal{C}$ calculées entre les différents algorithmes testés. Ce tableau indique par exemple que 93,91 % des solutions optimales trouvées par l'algorithme C sont dominées par les solutions optimales obtenues par l'algorithme B.

	Algorithme A	Algorithme B	Algorithme C	Algorithme D
Algorithme A		0 %	0 %	0 %
Algorithme B	100 %		2,97 %	87,09 %
Algorithme C	100 %	93,91 %		56,99 %
Algorithme D	100 %	0 %	2,97 %	

TABLE 5.5 – *Métrique $\mathcal{C}$ pour le problème du shelter 2D avec couloir*

Le tableau 5.5 complète l'analyse faite via la mesure des hypervolumes relatifs car elle met en évidence le fait que les performances globales des solutions générées par l'algorithme A sont nettement moins bonnes que les performances des solutions trouvées par les autres algorithmes d'optimisation.

5.2.5 Résultats et prise de décision

Cette partie s'intéresse plus particulièrement aux solutions obtenues pour un des algorithmes testés sur ce problème d'agencement du shelter : l'algorithme D, avec $j_{max} = 10$, qui combine l'algorithme génétique avec le module OPEA (optimisation du placement des espaces d'accessibilité), le module SEPA (algorithme de séparation) et le module PERT (perturbation locale). Cet algorithme a été initialisé par 200 individus et a généré 100 populations d'individus.

5.2.5.1 Influence de la différence géométrique $\triangle g$

La sous-section précédente, relative à la comparaison des stratégies d'optimisation, a montré que l'algorithme D, avec $j_{max} = 10$, a généré 18528 solutions admissibles, soit 92,64 % des solutions calculées par l'algorithme d'optimisation. Afin de prendre une décision, en termes de choix de conception, le concepteur doit dans un premier temps faire un tri parmi ce grand nombre de solutions. Afin d'effectuer ce tri, le concepteur peut détecter les variantes admissibles et les variantes admissibles Pareto-optimales par exemple. La détection des variantes dépend de deux principaux paramètres : $\triangle g$, la différence géométrique et $\triangle f$, la différence fonctionnelle. Ces deux paramètres sont expliqués plus en détails dans le chapitre 4.

Le tableau 5.6 montre l'évolution du nombre de variantes admissibles et de variantes admissibles Pareto-optimales en fonction du paramètre $\triangle g$. Le paramètre $\triangle f$ reste lui fixé à $10\,cm$. Dans ce même tableau est indiqué la proportion de solutions par rapport à la population de départ, c'est-à-dire l'ensemble des solutions admissibles pour les variantes admissibles et l'ensemble des variantes admissibles pour les variantes admissibles Pareto-optimales.

$\triangle g$ (mm)	Variantes admissibles	Variantes admissibles Pareto-optimales
0	18528 (100 %)	270 (1,46 %)
50	777 (4,19 %)	9 (1,16 %)
500	207 (1,11 %)	3 (1,45 %)
1000	149 (0,80 %)	3 (2,01 %)

TABLE 5.6 – *Shelter 2D avec couloir : évolution du nombre de variantes et variantes Pareto-optimales en fonction de* $\triangle g$

5.2.5.2 Exploration perceptive des solutions

Nous supposons maintenant que le concepteur va faire un choix parmi les variantes détectées avec une différence géométrique $\triangle g$ égale à $50\,cm$ (207 solutions). Il peut restreindre son choix aux solutions Pareto-optimales, qui réalisent le meilleur compromis entre tous les critères d'optimisation quantitatifs du problème d'optimisation. Il peut également élargir son choix à l'ensemble des

variantes admissibles. Il est possible de visualiser ces variantes sur un nuage de points en deux dimensions, où chaque axe représente un des deux objectifs. Pour rappel, les deux critères d'optimisation du problème d'agencement sont $\min F1$ et $\max F2$. Ce nuage de points est illustré sur la figure 5.6. Comme le montre cette figure, parmi ces 207 solutions, un certain nombre de solutions est plus ou moins éloigné de la surface de compromis du problème d'optimisation (points rouges sur la figure 5.6). Le concepteur a alors la possibilité de sélectionner une zone de points à explorer, localisée près de la surface de compromis. Par exemple, comme le montre la figure 5.6, le concepteur peut sélectionner les solutions qui vérifient les conditions suivantes sur les objectifs : $F1 \leqslant 10\,cm$ et $F2 \geqslant 450\,cm$. L'ensemble de solutions sélectionnées par le concepteur est alors composé de 66 solutions qui peuvent être ensuite explorées perceptivement. Le concepteur peut également sélectionner d'autres solutions en fonction de leur rang.

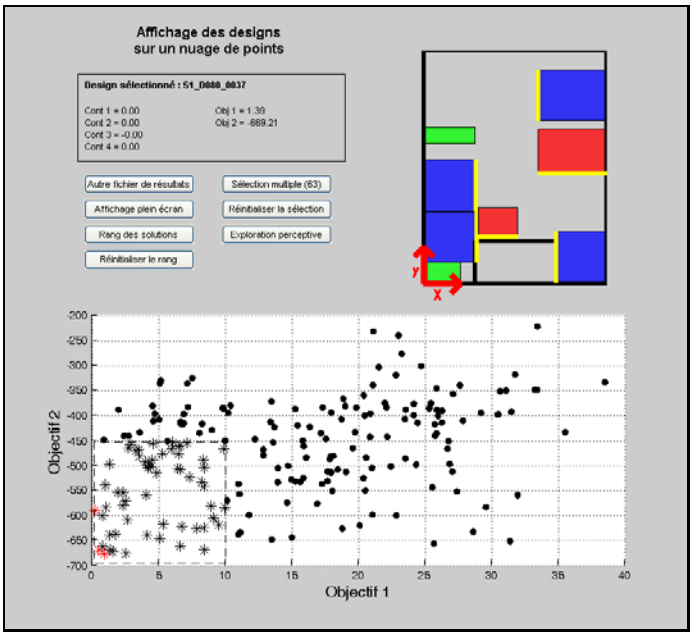


FIGURE 5.6 – *Nuage de points des 207 variantes admissibles*

Le chapitre 4 de ce manuscrit détaille une méthode interactive permettant l'exploration perceptive d'un ensemble de solutions d'agencement. Nous allons ici appliquer cette méthode sur l'ensemble des variantes admissibles générées par l'algorithme D ($j_{max} = 10$), avec $\triangle_g = 50\,cm$, soit 207 solutions. La méthode proposée consiste à présenter une population de 8 solutions au concepteur afin que celui-ci choisisse parmi ces solutions, celles qui correspondent le mieux à ses critères perceptifs. Ensuite, la méthode, qui utilise un algorithme génétique interactif propose au concepteur d'autres solutions jusqu'à converger vers une ou plusieurs solutions répondant au jugement personnel du concepteur.

Afin de démontrer l'efficacité de cette méthode, cette dernière est testée en mode automatique. Parmi toutes les solutions à explorer, une solution est considérée comme étant celle qui convient le

mieux aux critères perceptifs du concepteur. Cette solution est choisie aléatoirement. L'objectif du test de la méthode d'exploration perceptive en mode automatique est de retrouver cette solution, appelée solution « cible », en un minimum de générations. Pour cela, pour chaque population de l'algorithme composé de 8 individus, la sélection manuelle du concepteur est remplacée par une sélection automatique qui choisit systématiquement les deux individus les plus proches de la cible. La notion de proximité peut être définie suivant trois types de distance :

– la distance calculée par la norme 1 : $d_{ind-cible} = \sum_{i=1}^{n} |X_{ind}(i) - X_{cible}(i)|$
– la distance euclidienne (ou norme 2) : $d_{ind-cible} = \sqrt{\sum_{i=1}^{n} (X_{ind}(i) - X_{cible}(i))^2}$
– la distance calculée par la norme infinie : $d_{ind-cible} = max_{i=1}^{n} |X_{ind}(i) - X_{cible}(i)|$

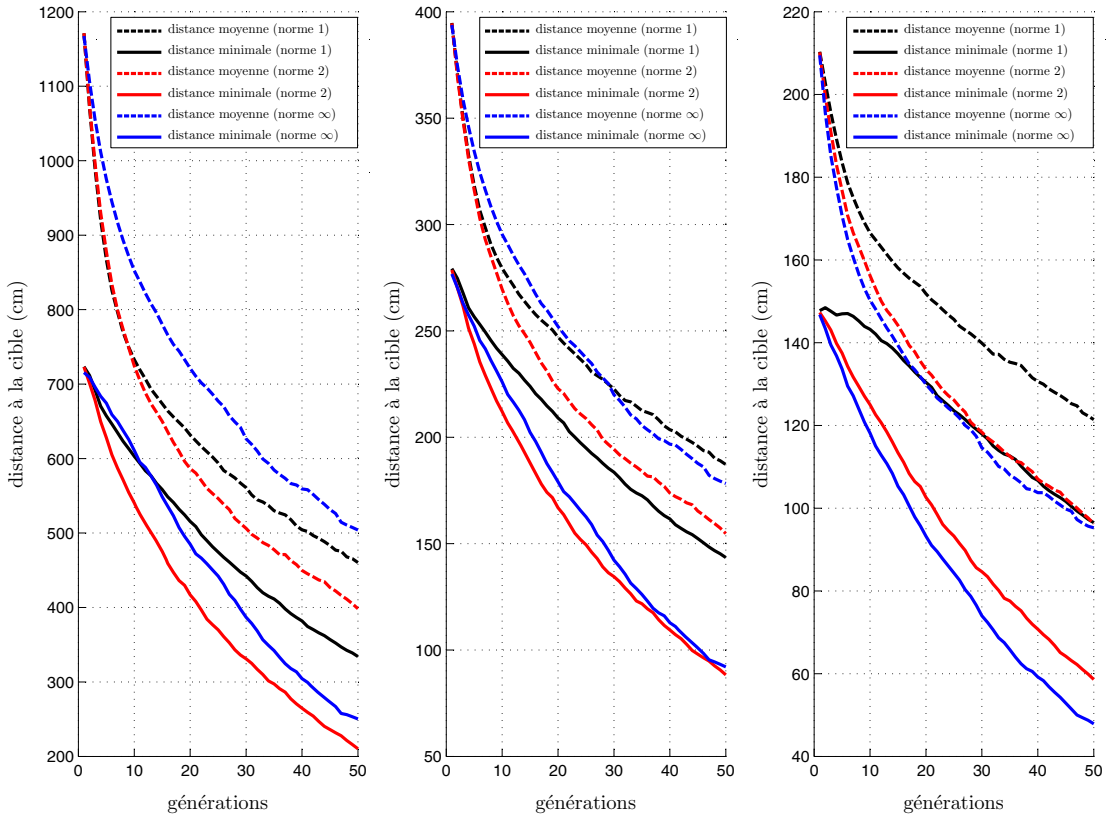
où X_{ind} définit le vecteur des variables d'optimisation pour l'agencement « ind » et X_{cible}, ce même vecteur pour l'agencement cible. Le paramètre n définit le nombre de variables d'optimisation.

Pour chaque type de norme (norme 1, norme euclidienne et norme infinie), le processus d'exploration perceptive automatique est réalisée sur 50 solutions cibles aléatoirement choisies parmi les solutions à explorer. Ce processus est réitéré 50 fois pour chaque solution cible, de manière à prendre en compte le comportement stochastique de l'algorithme génétique interactif. Le nombre maximal d'itérations de l'algorithme est fixé à 50. Le coefficient de roulette est fixé à 10, le coefficient de mutation à 0,1 et le coefficient de croisement à 0,8.

Suite aux différents calculs effectués, il convient d'analyser les résultats et de comparer la convergence de l'algorithme d'exploration pour les différentes normes testées. La figure 5.7 représente les trois graphes de convergence des trois processus d'exploration correspondant aux trois normes testées. Chaque graphe correspond à une norme de comparaison. Les différentes couleurs utilisées correspondent aux trois normes de convergence utilisées. Les courbes en traits pointillés représentent l'évolution de la distance moyenne de la population de solutions à la cible en fonction du nombre de générations de l'algorithme. Les autres courbes représentent la distance minimale à la cible en fonction du nombre de générations de l'algorithme. Pour une population donnée, la distance minimale définit la distance de l'individu le plus proche de la cible, suivant la norme de convergence utilisée.

Les courbes, représentées sur la figure 5.7 ont été construites en moyennant les résultats obtenus sur les $2500 = 50 \times 50$ simulations effectuées pour chaque norme de convergence testée. On remarque tout d'abord que l'algorithme génétique interactif converge car, pour chaque norme de convergence, la distance moyenne et la distance minimale à la cible diminuent continuellement. L'analyse des résultats met en relief le fait que la convergence de l'IGA est moins significative en utilisant la norme 1. En effet, pour cette norme, la distance minimale à la cible est la plus élevée à la fin du processus d'optimisation de l'algorithme. Par ailleurs, les courbes présentées sur la figure 5.7 montrent que les convergences par les normes 2 et 3 apportent des résultats équivalents.

Considérons par exemple les résultats issus de l'utilisation de la norme infinie comme norme de convergence de l'algorithme (figure 5.7 (c)). On remarque qu'à la fin du processus d'exploration, soit 50 générations de l'algorithme, la distance minimale n'est pas égale à 0. Ceci signifie donc que l'algorithme n'a pas trouvé exactement la solution cible. Cependant, la distance entre cette solution



(a) comparaison des convergences par la norme 1

(b) comparaison des convergences par la norme 2

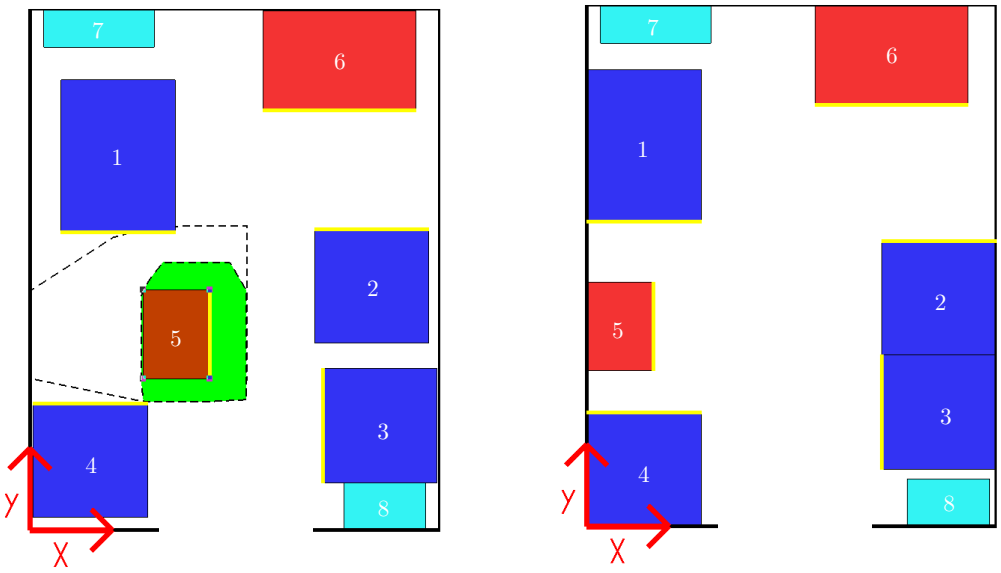
(c) comparaison des convergences par la norme infinie

FIGURE 5.7 – *Shelter 2D avec couloir : convergence de l'algorithme génétique interactif*

cible et la solution la plus proche, trouvée par l'algorithme est égale en moyenne à 50 cm (norme infinie). Aussi, l'analyse des résultats montre qu'en moyenne, pour la norme infinie, l'algorithme génétique interactif nécessite 20 générations pour identifier la solution cible ou une solution proche de celle-ci. Ce nombre de générations est raisonnable pour une utilisation manuelle de la méthode par le concepteur.

5.2.5.3 Modification manuelle et locale d'une solution

Considérons maintenant que le concepteur sélectionne une solution parmi les solutions proposées par l'algorithme d'optimisation, via la stratégie d'exploration perceptive, testée dans la partie précédente. Le concepteur a alors la possibilité de visualiser le modèle géométrique de cette solution, ainsi que les contraintes et les objectifs liés au problème d'optimisation d'agencement. Via l'interface interactive, le concepteur peut modifier manuellement et localement certains composants, afin d'adapter la solution générée par l'algorithme à sa propre expertise du problème d'agencement. Un exemple de modification locale d'une solution est illustré sur la figure 5.8. La figure 5.8 (a) représente la solution proposée par l'algorithme d'optimisation et la figure 5.8 (b) représente la solution modifiée par le concepteur.



(a) Solution proposée par l'algorithme (b) Solution modifiée par le concepteur

FIGURE 5.8 – *Shelter 2D avec couloir : modification manuelle et locale d'une solution*

Sur la figure 5.8 (a), on remarque que l'indicateur d'aide à la modification d'une solution a été utilisé en sélectionnant le bureau 1 (composant n°5). Cet indicateur est décrit dans le chapitre 4. Le concepteur a pu déplacer certains composants et avoir directement un retour sur la valeur des contraintes et des objectifs du problème d'optimisation. Finalement, comme l'illustre le tableau 5.7, la solution modifiée par le concepteur est aussi performante que la solution proposée par l'algorithme d'optimisation. Par ailleurs, cette solution modifiée inclut en plus l'expertise et le jugement personnel du concepteur.

Objectifs	Solution proposée	Solution modifiée
F1 (cm,min)	0,31	3,01
F2 (cm,max)	589,48	625,35

TABLE 5.7 – *Shelter 2D : comparaison des objectifs d'optimisation sur deux solutions (solution proposée vs solution modifiée)*

5.2.6 Modification du problème d'agencement : suppression du couloir central

5.2.6.1 Description et formulation du problème

Par rapport au problème d'agencement du shelter 2D avec couloir, décrit précédemment, une modification est apportée à la description du problème. Le couloir central n'est plus pris en compte. En effet, dans le premier cas d'étude, le couloir central a été introduit par le concepteur pour garantir l'accessibilité aux équipements depuis l'entrée du shelter. Modifier la description du problème pour prendre en compte toutes les exigences du concepteur est la solution la plus simple et la plus effi-

cace car elle ne complexifie pas le problème d'optimisation. Cependant, cette alternative restreint la recherche de solutions optimales vers une zone de solutions qui possèdent toutes un couloir central. Elle ne permet pas la recherche de solutions innovantes, non initialement pensées par le concepteur. Par exemple, dans le cas du shelter avec un couloir central, il est impossible d'obtenir des solutions proposant un îlot central regroupant certains équipements et laissant l'accès aux autres équipements.

Pour ce problème d'agencement du shelter, tous les équipements n'ont pas spécifiquement besoin d'être accessibles depuis l'entrée du shelter (seulement les armoires 1, 2, 3 et 4 et le bureau 2). En effet, les deux boîtiers n'ont pas besoin d'être accessibles et le bureau 1, qui une tablette pliante, est considéré comme toujours accessible.

La suppression du couloir central entraîne une modification de l'indice de faisabilité du problème d'agencement. Cet indice est maintenant égal à 57,14 % et est inférieur à l'indice de faisabilité du précédent problème d'agencement du shelter avec un couloir central (66,26 %).

Aussi, une seconde modification est apportée à la formulation du problème. En effet, le couloir central n'est plus pris en compte dans la description du problème mais il faut toujours garantir l'accessibilité à certains équipements du shelter. L'idée est donc de traduire cette exigence d'accessibilité aux équipements en un objectif du problème d'optimisation. Cette traduction est détaillée dans le chapitre 2, dans la partie évoquant la formulation d'un problème d'optimisation d'agencement.

Finalement, la description et la formulation du problème d'optimisation d'agencement du shelter sans couloir sont identiques à celles du shelter avec couloir, à deux détails près :
- le couloir central n'est plus pris en compte dans le modèle géométrique 2D du shelter,
- un troisième objectif est ajouté : **minimiser** la fonction accessibilité à certains composants depuis l'entrée du shelter.

Pour mesurer l'accessibilité aux composants depuis l'entrée du shelter, nous utilisons la méthode qui s'appuie sur le calcul du polygone d'appartenance entre le shelter et les équipements et le cercle qui modélise le concepteur. Le polygone définissant le shelter et les équipements est composé du shelter, des 4 armoires, du bureau 2 et des 2 boîtiers. Le cercle modélisant le concepteur a un rayon fixé à $50\,cm$.

5.2.6.2 Résolution du problème d'optimisation et résultats

Afin de résoudre ce problème d'optimisation d'agencement, nous décidons d'utiliser l'algorithme qui résulte de la combinaison entre l'algorithme génétique, le module OPEA (optimisation du placement des espaces d'accessibilité) et le module SEPA (algorithme de séparation). Cet algorithme est identifié par l'algorithme C dans la section précédente. Cet algorithme a été initialisé par 200 individus, tous aléatoirement créés. Le nombre maximal de générations de l'algorithme est fixé à 100. Les coefficients des opérateurs de croisement et de mutation pour les variables continues et discrètes sont respectivement fixés à 0,7 et 0,1. On réitère dix fois le processus de l'algorithme d'optimisation

pour prendre en compte le comportement stochastique de l'algorithme génétique.

A la fin du processus, l'algorithme d'optimisation a trouvé 16107 solutions admissibles, soit 80,54 % des solutions générées par l'algorithme. Les solutions admissibles sont ici considérées comme les solutions qui respectent toutes les contraintes du problème d'optimisation, avec $\triangle_c = 0\,cm^2$ pour chaque contrainte de non-chevauchement. Il est possible ensuite de détecter, parmi ces solutions admissibles, les variantes (par exemple avec $\triangle_g = 50\,cm$ et $\triangle_f = 10\,cm$) et les variantes Pareto-optimales. Ainsi, l'algorithme a généré 272 variantes admissibles et 21 variantes admissibles Pareto-optimales.

Ces variante admissibles Pareto-optimales peuvent être visualisées en utilisant le graphe parallèle, décrit dans le chapitre 4. Ce graphe parallèle est construit en prenant en compte toutes les contraintes et tous les objectifs du problème d'optimisation. La figure 5.9 illustre ce graphe parallèle. Un filtre est appliqué sur l'objectif d'accessibilité, afin de privilégier plus particulièrement les solutions qui garantissent au mieux l'accessibilité aux équipements depuis l'entrée du shelter.

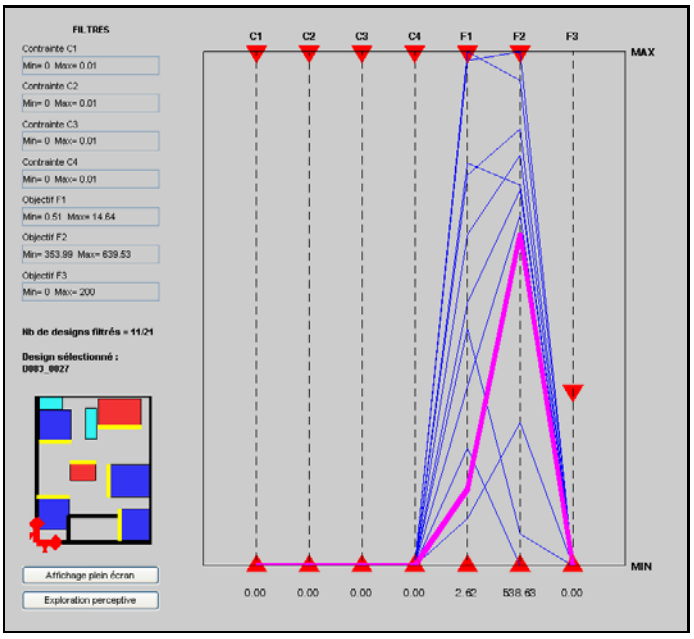


FIGURE 5.9 – *Problème d'agencement du shelter sans couloir : graphe parallèle des variantes admissibles Pareto-optimales*

La figure 5.10 illustre deux variantes admissibles Pareto-optimales qui garantissent l'accessibilité aux armoires et au bureau 2. Le polygone d'appartenance, montrant les zones où peut aller le concepteur dans le shelter, est indiqué en trait pointillé sur la figure. On remarque, sur cette figure, que les solutions proposées par l'algorithme d'optimisation se rapprochent des solutions d'agencement qui introduisent un couloir au centre du shelter.

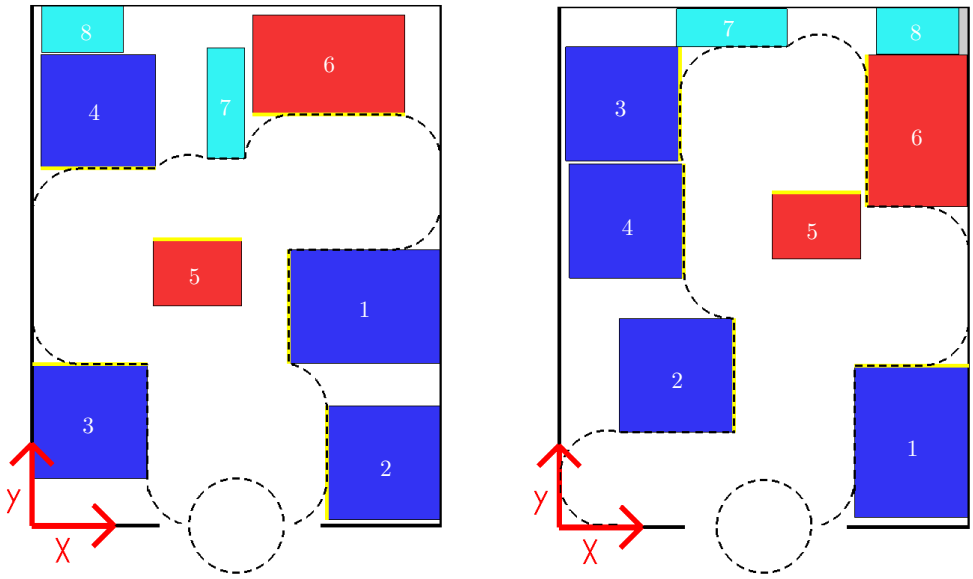


FIGURE 5.10 – *Shelter 2D sans couloir : variantes admissibles Pareto-optimales*

5.3 Agencement d'un shelter en 3D avec dimensions variables

Ce cas d'étude est une extension en trois dimensions du problème d'agencement du shelter, traité dans la section précédente. Cette section détaille le processus d'optimisation mis en œuvre pour ce problème d'agencement d'espace en 3D.

5.3.1 Description du problème

Par rapport au premier problème d'agencement, un coffre de rangement et un boîtier électrique sont ajoutés. Les dimensions et la masse de chaque équipement sont rappelés dans le tableau 5.8. Le coffre de rangement possède un espace d'accessibilité qui permet la manipulation de ce coffre. Les espaces libres (espace devant la porte, espace sous le climatiseur et couloir central), définis dans le premier problème d'agencement du shelter sont aussi pris en compte dans ce problème 3D. Leurs dimensions sont indiquées dans le tableau 5.2. Ce problème d'agencement comporte donc 10 composants matériels et 10 composants virtuels (7 espaces d'accessibilité et 3 espaces libres).

La figure 5.11 illustre le modèle géométrique du shelter en trois dimensions. Le numéro de chaque composant correspond à celui inscrit dans le tableau 5.8. Sur cette figure, les espaces libres ne sont pas représentés. Les climatiseurs, positionnés à l'arrière du shelter, sont eux représentés et portent les numéros 11 et 12. Le positionnement de ces deux climatiseurs est fixe.

N°	Contenant/Composant	Dim (mm)	Masse (kg)
0	Shelter	2150×2740×1910	
1	Armoire 1	800×600×1610	400
2	Armoire 2	600×600×1610	300
3	Armoire 3	600×600×1610	300
4	Armoire 4	600×600×1610	300
5	Bureau 1	350×465×50	10
6	Bureau 2	525×800×750	30
7	Coffre de rangement	365×1025×600	30
8	Boîtier 1	200×580×800	35
9	Boîtier 2	400×200×400	15
10	Boîtier 3	250×430×1185	35

TABLE 5.8 – *Shelter 3D : dimensions et masse du shelter et des équipements*

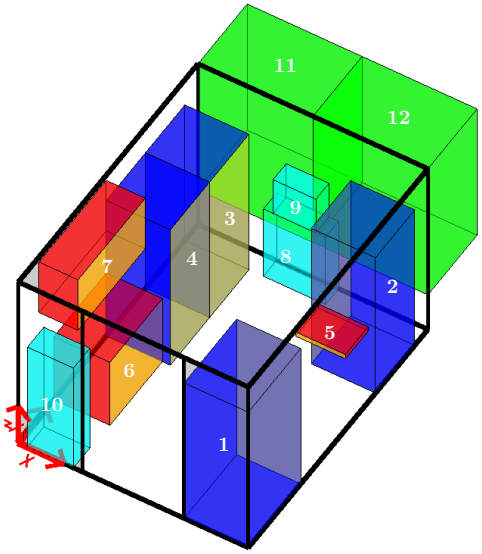


FIGURE 5.11 – *Modèle géométrique 3D du problème d'agencement du shelter*

5.3.2 Formulation du problème

Dans cette étape de la méthode d'optimisation, il convient de définir les variables d'optimisation, les contraintes et les objectifs du problème.

5.3.2.1 Variables de conception

Selon les exigences du concepteur, ces variables sont définies de la manière suivante :

- 2 variables continues (X et Y) et 2 variables discrètes (α et λ) pour la position, l'orientation et le sens de chaque armoire (4) et de chaque bureau (2),
- 1 variable continue (Y) et 1 variable discrète (X) pour la position du coffre de rangement (1),

– 1 variable continue (X) et 1 variable discrète (Y) pour la position du boîtier 1 (1) et du boîtier 3 (1),
– 2 variables continues (X et Z) et 1 variable discrète (Y) pour la position du boîtier 2 (1).

Les variables continues X, Y et Z varient respectivement de 0 à 2150 mm, de 0 à 2740 mm et de 0 à 1910 mm, correspondant aux dimensions du shelter. Étant donnée la forme parallélépipédique des composants, α peut prendre deux valeurs 0 et 90° (ou 1 et 6 selon la représentation adoptée dans cette approche et illustrée sur la figure 2.3), correspondant aux deux orientations possibles des composants dans le plan $(0, \vec{X}, \overrightarrow{Y})$. La variable λ est seulement définie pour les armoires et les bureaux qui possèdent un espace d'accessibilité. Étant donnée une orientation α du composant, il y a deux sens λ possibles : 1 ou 2.

Le positionnement du coffre de rangement est soumis à des exigences particulières. En effet, le concepteur souhaite que celui-ci soit fixé sur une des parois du shelter, positionnées en $X = 0$ ou $X = 2150$. La position sur l'axe $(O, \overrightarrow{Z})$ du coffre est également fixée ($Z = 1550\,mm$). La position sur l'axe $(O, \overrightarrow{Y})$ du coffre est libre. Les boîtiers 1 et 3 sont soumis à des exigences similaires à celles respectées par le coffre de rangement. Ces boîtiers sont posés au sol et sont fixés sur une des parois, à l'avant ou à l'arrière du shelter. Le boîtier 2 est également fixé sur les mêmes parois du shelter, mais sa position suivant l'axe $(O, \overrightarrow{Z})$ est libre.

Aussi, la longueur suivant l'axe $(O, \overrightarrow{Y})$ du shelter est variable. Par conséquent, une variable d'optimisation supplémentaire est définie. Cette variable continue $L_{shelter}$ varie de 0 à 2740 mm. En résumé, ce problème d'optimisation comporte 34 variables combinant des variables continues et discrètes.

5.3.2.2 Contraintes de conception

Comme pour le problème d'agencement du shelter en deux dimensions, les contraintes définies par le concepteur sont uniquement des contraintes de non-chevauchement entre composants. Ces contraintes sont également divisées en quatre catégories :
– C1 : non-chevauchement entre les composants matériels,
– C2 : non-chevauchement entre les composants matériels et les espaces d'accessibilité,
– C3 : non-chevauchement entre tous les composants et l'extérieur du shelter, l'espace devant la porte et le couloir central,
– C4 : non-chevauchement entre les armoires et l'espace sous l'entrée du climatiseur.

Il n'existe pas de contrainte de non-chevauchement entre les espaces d'accessibilité car on suppose que les opérations de chargement de matériel dans les armoires sont séquentielles et que deux personnes ne travaillent jamais simultanément dans le shelter. Deux espaces d'accessibilité peuvent donc se recouvrir. On suppose également que les espaces d'accessibilité peuvent chevaucher les différents espaces libres présents dans le shelter. Aussi, il existe une particularité pour le bureau 1. Ce bureau est une tablette qui peut se rabattre et on suppose donc qu'elle peut chevaucher tous

les composants virtuels présents dans le shelter. Étant donnée la forme parallélépipédique des composants, les contraintes de non-chevauchement sont évaluées via le calcul du volume d'intersection entre les composants. Le volume d'intersection entre les composants i et j est donné par la formule 2.2 (chapitre 2).

5.3.2.3 Objectifs d'optimisation

Les objectifs de ce problème d'optimisation sont similaires aux objectifs du problème d'agencement du shelter en deux dimensions. Un troisième objectif est ajouté, visant à réduire la longueur du shelter. Ces critères d'optimisation sont donc :

- **minimiser** (F1), avec F1 = distance entre le centre de gravité de l'ensemble des composants et le centre géométrique modifié du shelter,
- **maximiser** (F2), avec F2 = distance entre l'armoire 1 et l'ensemble composé des armoires 3 et 4 et du boîtier 2,
- **minimiser** (F3), avec F3 = longueur du shelter (variable $L_{shelter}$).

Toutes les distances, calculées dans l'évaluation de ces objectifs, sont définies par la norme euclidienne. Le centre géométrique modifié du shelter a pour coordonnées $(l_{shelter}, L_{shelter}, 0)$ où $l_{shelter}$ et $L_{shelter}$ définissent respectivement les dimensions du shelter suivant les $(0, \vec{X})$ et $(0, \vec{Y})$.

5.3.3 Indice de faisabilité du problème d'agencement

Le calcul de l'indice de faisabilité de ce problème d'agencement est semblable à celui effectué sur le problème du shelter en 2D. A la matrice d'intersections formulée pour le premier problème (équation 5.4), on ajoute 3 nouvelles lignes (et colonnes) pour traduire les contraintes de placement qui existent entre le coffre de rangement, son espace d'accessibilité, le boîtier n°2 et les autres composants de l'agencement. En appliquant la formule exprimant l'indice de faisabilité (équation 2.10), nous obtenons un indice égal à 60,52 %, ce qui montre à priori la faisabilité du problème d'optimisation d'agencement. Par ailleurs, si les composants de cet agencement avaient tous été considérés comme des composants matériels et si nous avions appliqué la formulation traditionnelle de la compacité, la compacité du problème serait égal à 85,14 %, démontrant à priori également la faisabilité du problème d'agencement.

5.3.4 Résolution du problème et résultats

Afin de résoudre ce problème d'optimisation multiobjectif, défini précédemment, l'algorithme génétique Omni-Optimizer, sans module ajouté, est utilisé. L'algorithme est initialisé par 200 individus tous aléatoirement générés. Les coefficients des opérateurs de croisement et de mutation pour les variables continues et discrètes sont respectivement fixés à 0,7 et 0,1. Le nombre d'itérations maximal autorisé pour la génération d'individus est fixé à 100. On réitère dix fois le processus de l'algorithme

génétique Omni-Optimizer pour prendre en compte le comportement stochastique de l'algorithme.

Comme pour le premier problème d'agencement du shelter, l'algorithme génétique seul n'a pas pu trouver une solution admissible au problème. En fixant le seuil de respect des contraintes à $\triangle_c = 0\,cm^2$, on remarque que toutes les contraintes de conception ne sont pas respectées. Par conséquent, nous décidons de combiner l'algorithme génétique avec plusieurs modules d'optimisation, présentés dans le chapitre 3. Deux algorithmes d'optimisation ont donc été testés sur ce problème d'optimisation :

- algorithme A : algorithme génétique + module OPEA + module SEPA
- algorithme B : algorithme génétique + module OPEA + module SEPA + module PERT (avec $j_{max} = 3$)

Le paramètre j_{max} est utilisé par le module d'optimisation PERT et représente ici le nombre maximal de tentatives de recherche d'une solution admissible. Le seuil d'activation des modules SEPA et PERT, correspondant à un seuil de respect des contraintes de placement, est fixé à $10\,dm^3$. Le nombre maximal d'itérations de l'algorithme d'optimisation utilisé dans le module SEPA est fixé à 100. Aussi, pour le module PERT, le nombre de composants dont l'orientation ou le sens change, est fixé à 2.

Afin de comparer les performances de ces deux algorithmes, ces derniers sont initialisés par la même population initiale que l'algorithme génétique. Les paramètres de chaque algorithme sont identiques à ceux de l'algorithme génétique, c'est-à-dire 0,7 pour le coefficient de croisement et 0,1 pour le coefficient de mutation, pour les variables continues et discrètes. On réitère également dix fois le processus de chaque algorithme pour prendre en compte le comportement stochastique des algorithmes. Le seul paramètre qui évolue entre les deux algorithmes d'optimisation est le nombre maximal d'itérations autorisé. En effet, parce que chaque algorithme n'évolue pas à la même vitesse et afin d'homogénéiser les résultats sur l'échelle du temps de calcul, la nombre maximal d'itérations pour l'algorithme A est fixé à 200 et à 100 pour l'algorithme B.

Comparaison quantitative

Comme précédemment, les performances des deux algorithmes d'optimisation sont comparés quantitativement, suivant le nombre de variantes admissibles générées au cours du temps. La figure 5.12 illustre cette comparaison. Afin de détecter les variantes, les paramètres $\triangle g$ (différence géométrique) et $\triangle f$ (différence fonctionnelle) sont respectivement fixés à $50\,cm$ et $100\,cm$.

Sur cette figure, on remarque que l'algorithme B génère plus de variantes admissibles que l'algorithme A. En effet, l'algorithme B a généré 148 variantes alors que l'algorithme A a créé 82 variantes (soit 80,5 % de variantes en plus). Ceci est dû à l'utilisation du module PERT qui facilite la recherche de solutions admissibles. Cependant, ce module PERT requiert un temps de calcul non négligeable et par conséquent, l'algorithme A trouve plus rapidement un certain nombre de variantes admissibles. En effet, après 30 min de calcul, l'algorithme A a déjà trouvé 17 variantes alors que l'algorithme B n'en a trouvé aucune. Comme pour le problème du shelter en 2D, le choix d'une stratégie d'op-

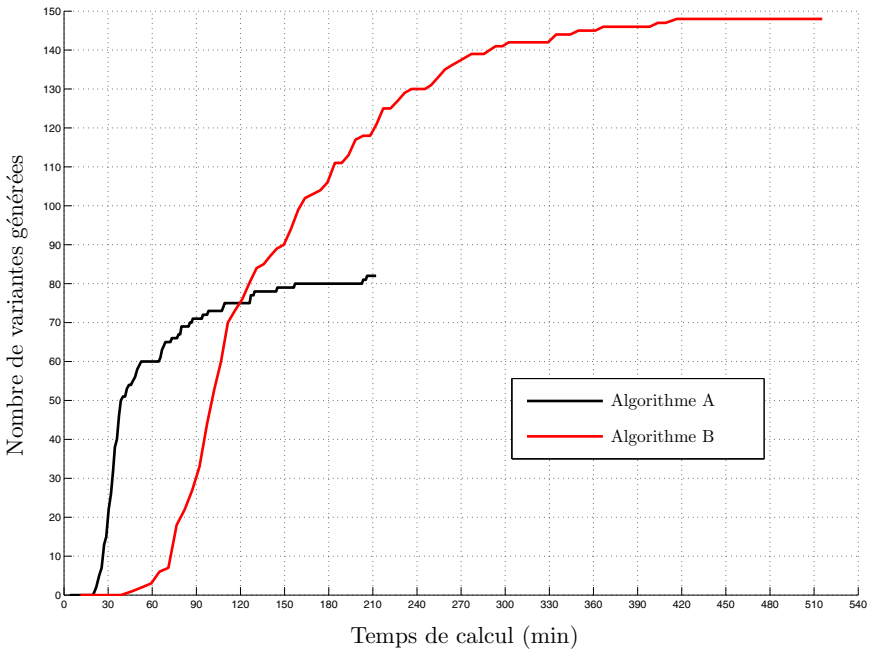


FIGURE 5.12 – *Problème d'agencement du shelter 3D avec couloir : comparaison quantitative des performances des algorithmes d'optimisation*

timisation la plus adaptée au problème résulte d'un compromis entre le nombre d'alternatives de conception trouvées et le temps de calcul nécessaire à la création de ces alternatives.

Comparaison qualitative

Afin de comparer qualitativement les performances des deux algorithmes testés, nous utilisons comme précédemment le calcul de l'hypervolume relatif (HVR) et de la métrique $\mathcal{C}$. Tout d'abord, nous regroupons l'ensemble des solutions admissibles Pareto-optimales générées par les deux algorithmes. Cet ensemble est composé de 5361 solutions. Parmi ces 5361 designs, 4374 solutions sont Pareto-optimales, soit 81,59 % des solutions. En considérant les objectifs formulés dans ce problème d'optimisation, nous définissons le point de référence $W = (10^4, 0, 10^4)$. L'hypervolume correspondant à ces solutions admissibles Pareto-optimales est égal à 52×10^9. Ensuite, en considérant le même point de référence, l'hypervolume des solutions trouvées par chaque algorithme est calculé. Ces hypervolumes sont ensuite normés par rapport à l'hypervolume de référence, égal à 52×10^9. Ces hypervolumes relatifs (HVR) sont regroupés dans le tableau 5.9.

Algorithmes	HVR
A	100 %
B	92,22 %

TABLE 5.9 – *Hypervolumes normés pour le problème du shelter 3D avec couloir*

Le tableau 5.9 montre que la surface de compromis (solutions admissibles Pareto-optimales) est de meilleur qualité pour l'algorithme A. Néanmoins, la valeur rapprochée de l'hypervolume relatif de l'algorithme B indique une surface de compromis de qualité très proche. Il est possible également de vérifier cette remarque via le calcul de la métrique $\mathcal{C}$, défini dans le chapitre 1. Le tableau 5.10 indique les différentes valeurs des métriques $\mathcal{C}$ calculées entre les deux algorithmes testés. Le tableau 5.10 confirme l'analyse faite via la mesure des hypervolumes relatifs. Il montre qu'un plus grand nombre de solutions trouvées par l'algorithme B sont dominées par les solutions de l'algorithme A.

	Algorithme A	Algorithme B
Algorithme A		80,51 %
Algorithme B	4,61 %	

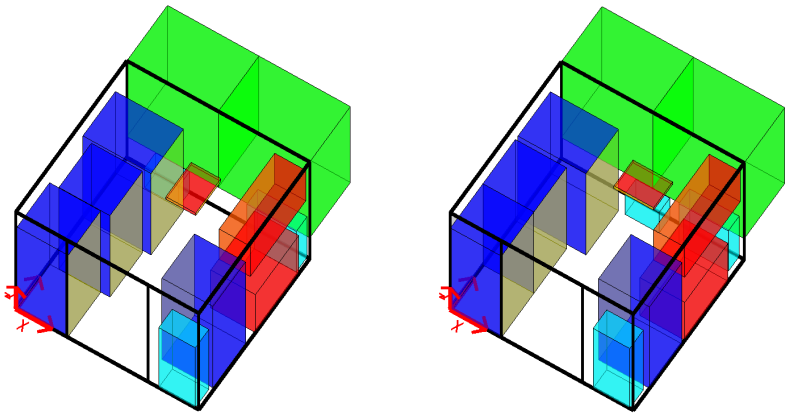
TABLE 5.10 – *Métrique $\mathcal{C}$ pour le problème du shelter 3D avec couloir*

Résultats

Considérons maintenant les variantes admissibles Pareto-optimales trouvées par l'algorithme A, qui combine l'algorithme génétique avec les modules d'optimisation OPEA et SEPA. Cet ensemble de solutions est composé de 15 solutions. Afin de faire un choix parmi ces alternatives de conception, il est possible de visualiser et comparer leurs performances via l'un des outils graphiques présentés dans le chapitre 4 : le nuage de points et le graphe parallèle. Supposons qu'à l'aide du graphe parallèle, le concepteur sélectionne la solution qui minimise la longueur du shelter (objectif F3). Cette solution est illustrée sur la figure 5.13 (a). Les valeurs des trois objectifs pour cette solution proposée par l'algorithme d'optimisation sont notées dans le tableau 5.11. Il possible alors pour le concepteur de modifier localement et manuellement la solution, afin d'y intégrer son propre jugement personnel sur la solution. La solution modifiée par le concepteur est illustrée sur la figure 5.13 (b). Le tableau 5.11 montre que la solution modifiée a des performances similaires à la solution proposée par l'algorithme. On remarque, sur les deux solutions illustrées sur la figure 5.11, la longueur du shelter est égale à $2046,10\,mm$, soit 25,32 % de réduction par rapport à la longueur initiale du shelter.

Objectifs	Solution proposée	Solution modifiée
F1 (cm,min)	91,66	91,89
F2 (cm,max)	501,66	505,34
F3 (mm,min)	2046,10	2046,10

TABLE 5.11 – *Shelter 3D : comparaison des objectifs d'optimisation sur deux solutions (solution proposée vs solution modifiée)*



(a) Solution proposée par l'algorithme (b) Solution modifiée par le concepteur

FIGURE 5.13 – *Shelter 3D avec couloir : modification manuelle et locale d'une solution*

5.4 Problématique de stockage de composants

Le problème d'agencement, décrit dans cette partie, porte sur la problématique de stockage de composants dans un contenant. Cette problématique de stockage est différente de la problématique de découpe et de conditionnement (« *cutting and packing problems* » en anglais), très largement traitée dans la littérature (WHS07). En effet, dans les applications de découpe et de conditionnement, deux types de problème peuvent être posés :

- problème de décision : tous les composants peuvent-ils être placés dans le contenant ?
- problème de minimisation / maximisation : combien au maximum de composants peuvent-être placés dans le contenant ?

La problématique de stockage, traité dans cette partie, reprend ces caractéristiques qui définissent uniquement des contraintes géométriques (collision entre composants et appartenance des composants au contenant). Le stockage de composants comprend également d'autres exigences fonctionnelles qui peuvent être exprimées par le concepteur (équilibre des masses, regroupement d'objets. . .). Ces exigences se retrouvent dans les applications industrielles de stockage telles le chargement d'une palette ou d'un camion.

Les sous-sections suivantes, en décrivant les étapes de la résolution du problème de stockage, valident ainsi l'utilisation de la méthode d'optimisation proposée dans ce manuscrit pour la résolution de ces problèmes d'agencement particuliers.

5.4.1 Description du problème

Ce problème de stockage de composants consiste à placer 30 objets, de forme parallélépipédique dans un contenant, lui aussi de forme parallélépipédique. Ces objets sont divisés en trois groupes de composants avec des dimensions identiques. La figure 5.14 illustre le modèle géométrique 3D de

ce problème d'agencement. Les dimensions du contenant et des composants sont indiqués dans le tableau 5.12. En résumé, ce problème d'agencement comporte 30 composants qui sont uniquement matériels.

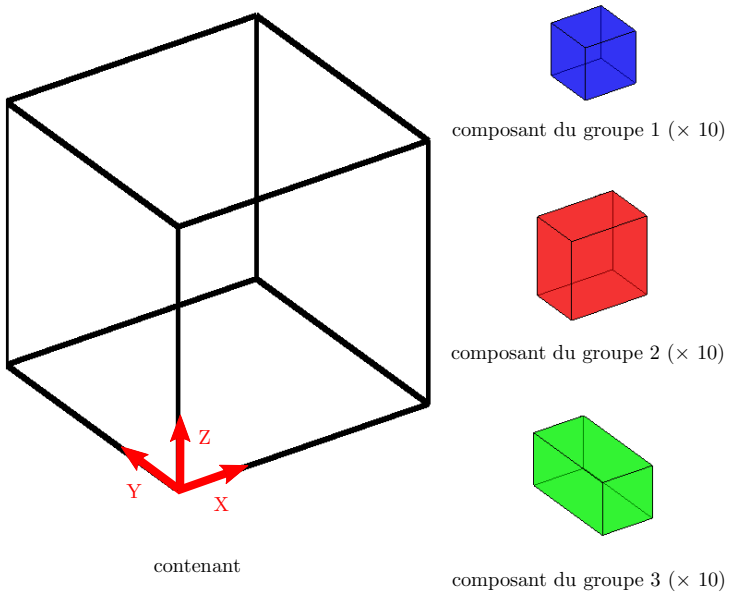


FIGURE 5.14 – *Modèle géométrique du problème de stockage de composants*

N°	Contenant/Composant	Dim (cm)	Masse (kg)
0	Contenant	100×100×100	
1	Composants groupe 1	20×20×20	10
2	Composants groupe 2	30×20×30	25
3	Composants groupe 3	40×20×20	20

TABLE 5.12 – *Stockage : dimensions et masse du contenant et des composants*

5.4.2 Formulation du problème

Dans cette étape de la méthode d'optimisation, il convient de définir les variables d'optimisation, les contraintes et les objectifs du problème.

5.4.2.1 Variables de conception

Selon les exigences du concepteur, ces variables sont définies de la manière suivante :
- 3 variables continues (X, Y et Z) pour la position des composants du groupe 1
- 3 variables continues (X, Y et Z) et 1 variables discrètes (α) pour la position et l'orientation des composants du groupe 2 et 3

Les variables continues X, Y et Z varient toutes de 0 à 100 cm correspondant à la dimension d'un côté du contenant. Étant donnée la forme cubique des composants du groupe 1, il n'y a pas plusieurs orientations possibles pour ces composants. Par ailleurs, pour les composants des groupes 2 et 3, la variable α, définissant l'orientation des composants, peut prendre deux valeurs possibles : 0 et 90° (ou 1 et 6 selon la représentation adoptée dans cette approche et illustrée sur la figure 2.3), correspondant aux deux orientations possibles des composants dans le plan $(0, \vec{X}, \overrightarrow{Y})$. En effet, on considère ici que pour ces composants, il y un côté « haut » et « bas » et les rotations des composants se font uniquement dans le plan $(0, \vec{X}, \overrightarrow{Y})$. En résumé, ce problème d'optimisation comporte 110 variables combinant des variables continues et discrètes.

5.4.2.2 Contraintes de conception

Comme pour le problème d'agencement du shelter, les contraintes définies par le concepteur sont uniquement des contraintes de non-chevauchement entre composants. Ces contraintes sont divisées en deux catégories :

- C1 : non-chevauchement entre les composants,
- C2 : non-chevauchement entre les composants et l'extérieur du contenant (appartenance des composants au contenant).

Étant donnée la forme parallélépipédique des composants, les contraintes de non-chevauchement sont évaluées via le calcul du volume d'intersection entre les composants. Le volume d'intersection entre les composants i et j est donné par la formule 2.2 (chapitre 2).

5.4.2.3 Objectifs d'optimisation

Nous supposons, pour ce problème de stockage, que les exigences exprimées par le concepteur sont multiples :

- équilibrer les masses à l'intérieur du contenant,
- minimiser la taille du contenant,
- regrouper au maximum les composants d'un même groupe.

Ce problème de stockage est donc un problème d'optimisation multiobjectif, à trois critères d'optimisation. Ces critères sont :

- **minimiser** (F1), avec F1 = distance entre le centre de gravité de l'ensemble des composants et le centre géométrique modifié du contenant,
- **minimiser** (F2), avec F2 = volume de la boîte englobante formée par l'ensemble des composants,
- **minimiser** (F3), avec F3 = somme des volumes des boîtes englobantes formées chacune par l'ensemble des composants d'un même groupe.

Le calcul de l'objectif F1 est similaire au calcul de l'objectif F1 pour le problème d'agencement du shelter. La distance entre le centre de gravité de l'ensemble des composants et le centre géométrique

modifié $(50, 50, 0)$ est évalué via la norme euclidienne. Afin de calculer les objectifs F2 et F3, on définit la notion de boite englobante de plusieurs composants comme étant le parallélépipède de taille minimale qui englobe ces composants. Considérons par exemple un ensemble de n composants. Chaque composant i est défini par la position de son centre géométrique (x_i, y_i, z_i) dans le repère $(O, \vec{X}, \overrightarrow{Y}, \overrightarrow{Z})$ et par ses dimensions (l_i, L_i, h_i) dans ce même repère. La boîte englobante Be de ces n composants est donc définie (pour $i \in \{1, ..., n\}$) par :

$$\begin{aligned} Be &= [\max(x_i + \frac{l_i}{2}) - \min(x_i - \frac{l_i}{2})] \\ &\quad \times [\max(y_i + \frac{L_i}{2}) - \min(y_i - \frac{L_i}{2})] \\ &\quad \times [\max(z_i + \frac{h_i}{2}) - \min(z_i - \frac{h_i}{2})] \end{aligned} \tag{5.5}$$

5.4.3 Indice de faisabilité du problème d'agencement

L'indice de faisabilité de ce problème de stockage correspond ici au calcul traditionnel de la compacité. En effet, tous les composants sont matériels et il n'existe pas de contraintes de non-chevauchement particulières entre les composants. L'indice de faisabilité (ou compacité) de ce problème d'agencement est égal à 42 %. Cette compacité, relativement faible signifie donc que le problème d'agencement peut à priori être résolu et qu'il y a assez de place dans le contenant afin d'y placer tous les composants. De même, l'objectif d'optimisation F2 traduisant l'exigence de minimisation de la taille du contenant peut également être interprété comme un objectif de maximisation de la compacité du problème d'agencement.

5.4.4 Résolution du problème et résultats

Afin de résoudre ce problème de stockage, nous décidons d'utiliser l'algorithme qui résulte de la combinaison entre l'algorithme génétique, le module SEPA (algorithme de séparation) et le module GRAV (prise en compte de la gravité). La première version du module GRAV (sans optimisation locale des surfaces de contact entre composants) est ici utilisée. Ces trois modules sont détaillés dans le chapitre 3. L' algorithme modulaire est initialisé par 600 individus, tous aléatoirement créés. Le nombre maximal de générations de l'algorithme est fixé à 100. Les coefficients des opérateurs de croisement et de mutation pour les variables continues et discrètes sont respectivement fixés à 0,7 et 0,1. On réitère dix fois le processus de l'algorithme d'optimisation pour prendre en compte le comportement stochastique de l'algorithme génétique.

A la fin du processus, l'algorithme d'optimisation a trouvé 56170 solutions admissibles, soit 93,62 % des solutions générées par l'algorithme. Les solutions admissibles sont ici considérées comme les solutions qui respectent toutes les contraintes du problème d'optimisation, avec $\triangle_c = 0\, cm^3$

pour chaque contrainte de non-chevauchement. Nous décidons de détecter directement, parmi ces solutions admissibles, les solutions admissibles Pareto-optimales, qui réalisent le meilleur compromis entre tous les critères d'optimisation. Cet ensemble de solutions est composé de 535 solutions.

Afin de comparer les performances de ces solutions admissibles Pareto-optimales, celles-ci sont affichées sur un graphe parallèle regroupant les valeurs des contraintes et des objectifs d'optimisation. Ce graphe parallèle est illustré sur la figure 5.15.

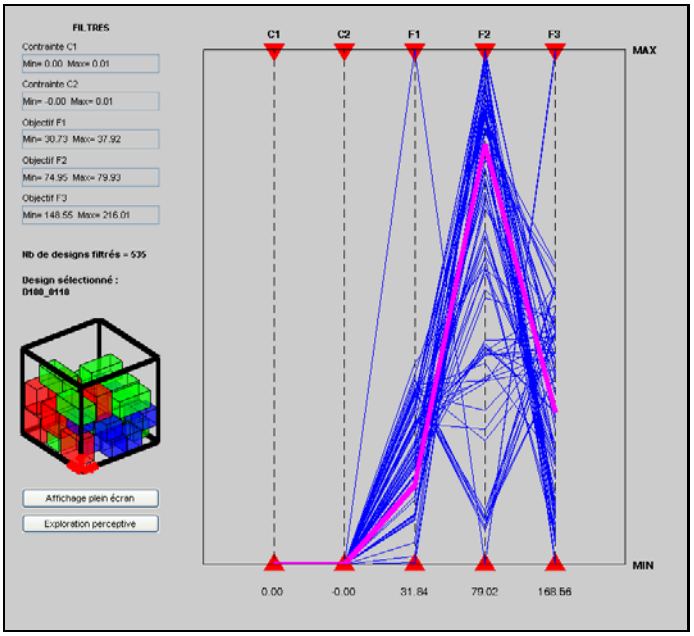


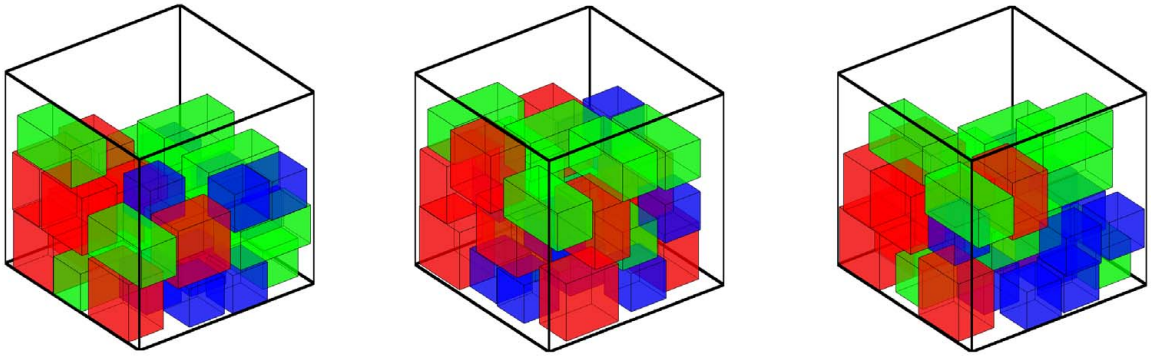
FIGURE 5.15 – *Problème de stockage : graphe parallèle des variantes admissibles Pareto-optimales*

Le tableau 5.13 indique par exemple trois solutions admissible Pareto-optimales. La solution n°1 minimise l'objectif F1, la solution n°2 minimise l'objectif F2 et la solution n°3 minimise l'objectif F3. L'unité de volume pour les objectifs F2 et F3 est en $10^4 cm^3$, ce qui permet de comparer ces volumes par rapport au volume du contenant égal à $100 \times 10^4 cm^3$. La figure 5.16 décrit les modèles géométriques de ces trois solutions (composants du groupe 1 en bleu, du groupe 2 en rouge et du groupe 3 en vert).

Solutions admissibles Pareto-optimales	Objectif F1 (cm,min)	Objectif F2 ($10^4 cm^3$,min)	Objectif F3 ($10^4 cm^3$,min)
Solution 1	**30,73**	79,34	177,00
Solution 2	37,92	**74,95**	215,42
Solution 3	33,07	79,63	**148,55**

TABLE 5.13 – *Problème de stockage : objectifs d'optimisation de 3 solutions*

.



(a) Solution n°1 (b) Solution n°2 (c) Solution n°3

FIGURE 5.16 – *Problème de stockage : solutions admissibles Pareto-optimales*

5.5 Démonstrateur d'optimisation d'agencement d'espace

Afin de matérialiser la méthode d'optimisation d'agencement, proposée dans ce manuscrit, un démonstrateur d'optimisation d'agencement d'espace a été développé au cours de ce doctorat. Ce démonstrateur a été développé via le logiciel de calcul numérique Matlab. L'objectif de ce démonstrateur est de proposer au concepteur un outil qui regroupe toutes les étapes de la méthode d'optimisation d'agencement : description, formulation, résolution du problème d'optimisation d'agencement et prise de décision.

La figure 5.17 illustre l'interface graphique principale du démonstrateur d'optimisation d'agencement. L'architecture du logiciel est divisée en 3 blocs qui ont les fonctionnalités suivantes :

- le bloc « **description et formulation du problème** » : permet de décrire le problème (nombre de composants, de contraintes, d'objectifs...) et définir les attributs de tous les composants (dimensions, degrés de liberté, bornes et base des variables de placement libres). Il est possible également de définir des variables d'optimisation supplémentaires (dimension par exemple) autres que les variables de positionnement des composants. La formulation des contraintes et des objectifs d'optimisation se fait directement par l'écriture de deux fichiers, écrits dans le langage propre au logiciel Matlab. Ces fichiers regroupent des fonctions, qui ont une structure bien définie et qui expriment les contraintes et les objectifs du problème d'optimisation.
- le bloc « **stratégie de résolution** » : permet tout d'abord de définir une population initiale pour l'algorithme (aléatoire ou choisie par le concepteur), sélectionner les modules d'optimisation en réglant leurs paramètres et enfin exécuter la stratégie modulaire associée.
- le bloc « **aide à la décision** » : regroupe tous les outils permettant au concepteur de trier les solutions générées par l'algorithme, les visualiser dans des espaces multidimensionnels, les explorer de manière perceptive et interagir avec une solution en particulier.

Actuellement, avec ce démonstrateur, il est possible de traiter seulement des problèmes d'optimisation d'agencement d'espace qui manipulent des composants de forme parallélépipédique.

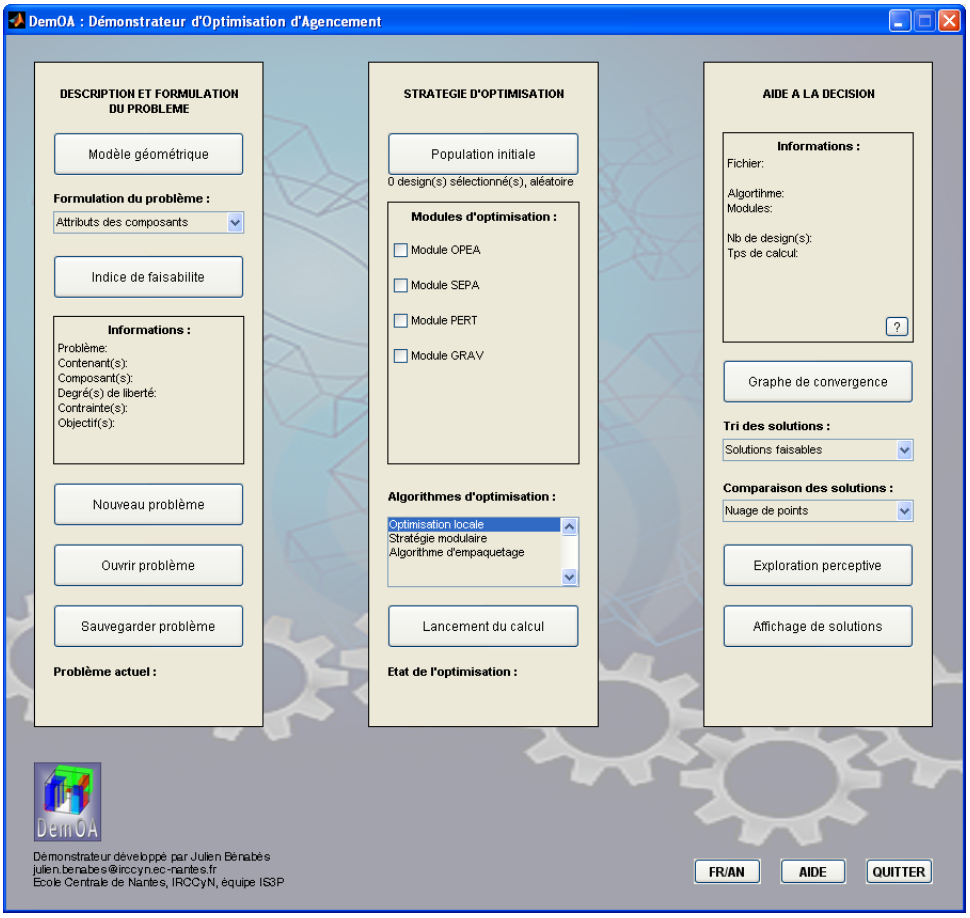


FIGURE 5.17 – *Interface graphique du démonstrateur d'optimisation d'agencement*

5.6 Conclusion

Ce chapitre a montré que la méthode d'optimisation d'agencement d'espace, proposée dans ce manuscrit, peut être appliquée à différentes applications industrielles. La méthode a été testée sur un problème d'agencement d'un shelter en deux et trois dimensions et sur un problème de stockage de composants dans un container. Le démonstrateur, qui supporte l'intégralité de la démarche, fournit un outil adapté à la résolution des problèmes d'optimisation d'agencement d'espace, manipulant des composants de forme parallélépipédique.

Conclusion générale et perspectives

La méthode de conception proposée dans ce manuscrit de thèse vise à résoudre de manière intégrée, générique et interactive un problème d'optimisation d'agencement d'espace. Cette méthode d'optimisation est structurée en quatre étapes qui guident le concepteur depuis la formulation de son besoin jusqu'à l'obtention d'une solution d'agencement idéale. Ces quatre étapes sont : la description, la formulation et la résolution du problème et la prise de décision finale.

La première étape consiste à créer un modèle géométrique de l'agencement étudié. Ce modèle géométrique est basé sur une décomposition innovante des composants en éléments matériels et virtuels. Cette décomposition permet notamment de prendre en compte le maximum de spécificités de chaque problème d'agencement. La seconde étape de formulation est dédiée à l'écriture du problème d'optimisation d'agencement, c'est-à-dire la définition des variables de conception, des contraintes et des critères d'optimisation. Cette étape est une étape de traduction des exigences exprimées par le concepteur en expressions mathématiques explicites. Certaines de ces exigences peuvent être difficilement intégrées dans la formulation du problème d'optimisation. C'est notamment le cas de la contrainte d'accessibilité aux composants depuis l'entrée du contenant. Deux méthodes sont proposées, dans ce manuscrit, afin de traduire cette exigence particulière en une contrainte ou un objectif d'optimisation. Lorsque les deux premières étapes de la méthode sont réalisées, il est possible alors d'évaluer la faisabilité du problème d'optimisation d'agencement. Un nouvel indice, proche du calcul traditionnel de la compacité d'un agencement, a été développé. Cet indice indique au concepteur si son problème peut à priori être résolu ou s'il n'existe aucune solution.

L'étape de résolution du problème d'optimisation d'agencement est basé sur l'utilisation d'une stratégie d'optimisation multiobjectif et modulaire. Le choix d'une approche multiobjectif permet d'axer les choix du concepteur sur les solutions optimales proposées par l'algorithme et non sur les critères d'optimisation, qui ne sont généralement pas tous maîtrisés et de même dimension. La modularité de l'approche permet d'avoir une stratégie d'optimisation à la fois générique et efficace, capable de s'adapter au plus grand nombre d'applications, tout en résolvant les problèmes dans des temps de calcul raisonnables. Cette stratégie d'optimisation est donc basée sur l'utilisation d'un algorithme génétique, couplé avec des modules d'optimisation locale. L'algorithme génétique a été privilégié car il s'adapte aux problèmes d'agencement les plus complexes, où la multiplication des contraintes morcellent l'espace de conception. Les modules d'optimisation aident l'algorithme génétique dans sa recherche de solutions optimales, en prenant en compte les spécificités de certains

problèmes. Dans ce manuscrit, quatre modules d'optimisation sont détaillés.

Enfin, la dernière étape du processus d'optimisation d'agencement repose sur la prise de décision finale du concepteur. Pour faire ce choix, l'approche décrite dans ce manuscrit propose au concepteur un ensemble de méthodes et d'outils graphiques qui lui permettent de visualiser et comparer les agencements dans un espace multidimensionnel. Aussi, une méthode d'exploration perceptive de solutions est présentée. Cette méthode, qui utilise un algorithme génétique interactif, permet de parcourir un ensemble de *designs*, en prenant en compte les perceptions et le jugement personnel du concepteur. L'objectif de cette approche est d'amener le concepteur à choisir une solution parmi toutes les solutions de compromis générées par l'algorithme. Ensuite, il peut, via une interface graphique interactive, associée à un outil de réalité virtuelle, visualiser et modifier localement une solution. Cette modification manuelle permet d'intégrer à la solution optimale proposée par l'algorithme l'expertise du concepteur.

Même si le travail réalisé dans ce doctorat apporte une réponse globale à la résolution des problèmes d'optimisation d'agencement d'espace, il reste quelques pistes de recherche à exploiter. Ces axes de recherche sont détaillés brièvement ici :

- créer un système expert permettant le recueil des exigences formulées par le concepteur et leur intégration dans la formulation du problème d'optimisation d'agencement. L'idée est d'avoir une base de données de fonctions mathématiques et d'outils de calcul qui modélisent le maximum d'exigences généralement exprimées dans les applications d'agencement d'espace. L'objectif à long terme est de rendre le concepteur autonome dans l'étape de description et de formulation du problème d'optimisation d'agencement. Un travail connexe peut notamment être consacré au recueil du besoin du concepteur et la hiérarchisation de ses exigences selon leur importance, afin de pouvoir déterminer le nombre de contraintes et d'objectifs d'optimisation nécessaires à la formulation du problème d'optimisation,
- adapter la méthode de calcul de l'accessibilité aux composants depuis l'entrée du contenant aux problèmes d'agencement en trois dimensions,
- améliorer l'évaluation de l'indice de faisabilité d'un problème d'agencement en développant de nouvelles stratégies d'optimisation. Ces stratégies d'optimisation doivent permettre de trouver de manière fiable l'espace minimal occupé par les composants qui peuvent chevaucher d'autres composants. Ceci permettra alors de donner au concepteur un indice de faisabilité du problème d'agencement qui soit encore plus précis que celui proposé dans ce manuscrit,
- multiplier les modules d'optimisation locale aidant l'algorithme génétique dans sa recherche de solutions optimales. Ces modules doivent permettre de prendre en compte les spécificités des problèmes d'agencement qui sont traités. Un travail pourra également être consacré à l'étude de l'influence de l'ordre d'intégration des modules dans la structure de l'algorithme génétique par rapport aux performances globales de l'algorithme d'optimisation. Aussi, ce même travail pourra évaluer les éventuelles influences qui peuvent exister entre ces modules d'optimisation,
- mettre en œuvre une typologie des problèmes d'optimisation d'agencement d'espace. Cette typologie ne doit pas seulement être un recueil des différentes applications d'agencement qui peuvent

exister mais doit servir de guide pour le concepteur dans le choix d'une stratégie d'optimisation la plus adaptée. Le concepteur pourra, lorsqu'il aura identifié son problème dans la typologie créée, choisir le ou les modules parmi ceux suggérés par la typologie,

- adapter la méthode aux problèmes d'agencement manipulant des composants de forme irrégulière. Cette adaptation n'a pas d'incidence sur le fonctionnement global du processus d'optimisation. Le travail consiste essentiellement à ajuster les variables qui définissent la position et l'orientation des composants dans l'espace et à adapter les méthodes de calcul des contraintes de non-chevauchement entre composants, notamment pour les problèmes 3D. Le calcul des gradients de ces contraintes de non-chevauchement (notamment pour l'utilisation du module d'optimisation SEPA) est aussi un point important à étudier,
- étendre les possibilités d'interaction entre la réalité virtuelle et l'optimisation d'agencement d'espace. Dans la méthode proposée dans ce manuscrit, une interface de réalité virtuelle utilisant un joystick à retour d'efforts a été implémentée. Cette interface est essentiellement utilisée pour la manipulation des composants dans les applications en 3D. L'objectif est maintenant de développer et tester d'autres interfaces de réalité virtuelle (salle immersive...) encore plus adpatées à la manipulation d'objets dans un epsace en trois dimensions. Aussi, un travail pourra être consacré à l'évaluation de l'apport de cette interface de réalité virtuelle pour l'optimisation d'agencement, en comparaison par exemple avec une manipulation simple à la souris des composants. Des tests avec des concepteurs placés dans les deux situations pourront être mis en place.

Références bibliographiques

[AA76] Adamowicz, M. and Albano, A. : Nesting two-dimensional shapes in rectangular modules. *Computer-Aided Design*, 8(1):27–33, janvier 1976.

[Aka00] Akao, Y. : *QFD prendre en compte les besoins du client dans la conception du produit*. 2000.

[ALC+04] Agrawal, G., Lewis, K., Chugh, K., Huang, C. H., and Bloebaum, C. L. : Intuitive visualization of pareto frontier for multi- objective optimization in n-dimensional performance space. *In Proceedings of the 10th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference, Albany, New York*, pages 2004–4434, 2004.

[APR65] Amerine, M. A., Pangborn, R. M., and Roessler, E. B. : *Principles of Sensory Evaluation of Food*. Academic Press, 1965.

[Bal99] Balling, R. : Design by shopping : A new paradigm. *In Proceedings of the Third World Congress of Structural and Multidisciplinary Optimization, Buffalo, NY, pp. 295–297.*, 1999.

[BBPR10a] Bénabès, J., Bennis, F., Poirson, E., and Ravaut, Y. : Accessibility in layout optimization. *In Proceedings of the Engineering Optimization (EngOpt) 2010 Conference, Lisbonne, Portugal*, 2010.

[BBPR10b] Bénabès, J., Bennis, F., Poirson, E., and Ravaut, Y. : An interactive-based approach to the layout design optimization. *In Proceedings of the 20th CIRP Design Conference, Nantes, France*, 2010.

[BBPR10c] Bénabès, J., Bennis, F., Poirson, E., and Ravaut, Y. : Interactive optimization strategies for layout problems. *International Journal on Interactive Design and Manufacturing*, 4:181–190, 2010.

[BBPR10d] Bénabès, J., Bennis, F., Poirson, E., and Ravaut, Y. : A new approach for specifying and solving layout problems. *In Proceedings of the IDMME-Virtual Concept Conference, Bordeaux France*, 2010.

[Bea85] Beasley, J. : An exact two-dimensional non-guillotine cutting tree search procedure. *Operations Research*, 33(1):49–64, 1985.

[BHKW07] Burke, E., Helier, R., Kendall, G., and Whitwell, G. : Complete and robust no-fit polygon generation for the irregular srock cutting problem. *European Journal of Operational Research*, 179:27–49, 2007.

[BKW04] Burke, E. K., Kendall, G., and Whitwell, G. : A new placement heuristic for the orthogonal stock-cutting problem. *OPERATIONS RESEARCH*, 52(4):655–671, 2004.

[BMN05] Birgin, E. G., Morabito, R., and Nishihara, F. H. : A note on an l-approach for solving the manufacturer's pallet loading problem. *Journal of the Operational Research Society*, 56:1448–1451, 2005.

[BP85] Brans, J. and P., V. : A preference ranking organisation method : The promethee method for mcdm. *Management Science*, 31:647–656, 1985.

[BPBR11a] Bénabès, J., Poirson, E., Bennis, F., and Ravaut, Y. : Démarche intégrée pour l'optimisation d'agencement d'espace : application à l'aménagement d'un shelter. *In Actes du 12ème Colloque National AIP PRIMECA, Mont-Dore, France*, 2011.

[BPBR11b] Bénabès, J., Poirson, E., Bennis, F., and Ravaut, Y. : Interactive modular optimization strategy for layout problems. *In Proceedings of the IDETC-CIE conference, Washington, USA*, 2011.

[BPBR11c] Bénabès, J., Poirson, E., Bennis, F., and Ravaut, Y. : Modular optimization strategy for layout problems. *In Proceedings of the International Conference on Engineering Design (ICED), Copenhague, Danemark*, 2011.

[Bro70] Broyden, C. G. : The convergence of a class of double-rank minimization algorithms. *Journal of the Institute of Mathematics and Its Applications*, 6(3):222–231, September 1970.

[BRT07] Brintrup, A. M., Ramsden, J., and Tiwari, A. : An interactive genetic algorithm-based framework for handling qualitative criteria in design optimization. *Computers in Industry*, 58:279–291, 2007.

[BS07] Bennell, J. A. and Song, X. : A beam search implementation for the irregular shape packing problem. *Journal of Heuristics*, 2007.

[CCD+96] Cagan, J., Clark, R., Dastidar, P., Szykman, S., and Weisser, P. : HVAC CAD layout tools : A case study of university/industry collaboration. *In ASME Design Engineering Technical Conferences and Computers in Engineering Conference*, 1996.

[CDY98] Chedmail, P., Damay, T., and Yannou, B. : A cooperative approach in mechanism design. *In Proceedings of the Integrated Design and Manufacturing in Mechanical Engineering (IDMME) Conference, Compiègne, France*, 1998.

[Coe00] Coello, C. A. : An updated survey of ga-based multiobjective optimization techniques. *ACM Comput. Surv.*, 32(2):109–143, 2000.

[CS02] Colette, Y. and Siarry, P. : *Optimisation multiobjectif*. 2002.

[CSY02] Cagan, J., Shimada, K., and Yin, S. : A survey of computational approaches to the three-dimensional layout problems. *Computer-Aided Design*, 34:597–611, 2002.

[CYD10] Cluzel, F., Yannou, B., and Dihlmann, M. : Evolutive design of car silhouettes using an interactive genetic algorithm. Rapport technique, Laboratoire Génie Industriel, Ecole Centrale Paris, October 2010. Cahiers de recherche 2010-22.

[DAPM00] Deb, K., Agrawal, S., Pratap, A., and Meyarivan, T. : A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization : Nsga-ii. pages 849–858. Springer, 2000.

[DD92] Dowsland, K. A. and Dowsland, W. B. : Packing problems. *European Journal of Operational Research*, 56:2–14, 1992.

[Deb98] Deb, K. : Multi-objective genetic algorithms : Problem difficulties and construction of test problems. *Evolutionary Computation*, 7:205–230, 1998.

[DGF11] Dong, H., Guarneri, P., and Fadel, G. : Bi-level approach to vehicle component layout with shape morphing. *Journal of Mechanical Design*, vol. 133, 2011.

[dNE05] do Nascimento, H. A. and Eades, P. : User hints : a framework for interactive optimization. *Future Generation Computer Systems*, 21:1177–1191, 2005.

[DPAM02] Deb, K., Pratap, A., Agarwal, S., and Meyarivan, T. : A fast and elitist multiobjective genetic algorithm : Nsga-ii. *IEEE Transactions on Evolutionary Computation*, 6:182–197, 2002.

[DPHG07] Drira, A., Pierreval, H., and Hajri-Gabouj, S. : Facility layout problems : A survey. *Annual Reviews in Control*, 31:255–267, 2007.

[DT08] Deb, K. and Tiwari, S. : Omni-optimizer : A generic evolutionary algorithm for single and multi-objective optimization. *European Journal of Operational Research*, 185(3):1062–1087, March 2008.

[DTR06] Dean, H. T., Tu, Y., and Raffensperger, J. F. : An improved method for calculating the no-fit polygon. *Computer and Operations Research*, 33(6):1521–1539, 2006.

[Dyc90] Dyckhoff, H. : A typology of cutting and packing problems. *European Journal of Operational Research*, 44(2):145–159, January 1990.

[Ege03] Egeblad, J. : Placement techniques for vlsi layout using sequence-pair legalization. Mémoire de D.E.A., University of Copenhagen, Department of Computer Science, July 2003.

[Ehr05] Ehrgott, M. : *Multicriteria Optimization.* Springer Berlin - Heidelberg, 2 édition, 2005.

[EL02] Eddy, J. and Lewis, K. E. : Visualization of multidimensional design and optimization data using cloud visualization. *ASME Conference Proceedings*, 2002(36223):899–908, 2002.

[EP09] Egeblad, J. and Pisinger, D. : Heuristic approaches for the two- and three-dimensional knapsack packing problem. *Comput. Oper. Res.*, 36:1026–1049, April 2009.

[EW07] Engau, A. and Wiecek, M. : 2d decision-making for multicriteria design optimization. *Structural and Multidisciplinary Optimization*, 34:301–315, 2007. 10.1007/s00158-006-0078-y.

[FF93] Fonseca, C. M. and Fleming, P. J. : Genetic algorithms for multiobjective optimization : Formulation, discussion and generalization. *In Genetic Algorithms : Proceedings of the Fifth International Conference*, pages 416–423. Morgan Kaufmann, 1993.

[Fle70] Fletcher, R. : A new approach to variable metric algorithms. *The Computer Journal*, 13(3): 317–322, 1970.

[Fuc06] Fuchs, P. : *Le Traité de la Réalité Virtuelle.* mars 2006.

[GDF72] Geoffrion, A. M., Dyer, J. S., and Feinberg, A. : An interactive approach for multi-criterion optimization, with an application to the operation of an academic department. *MANAGEMENT SCIENCE*, 19(4-Part-1):357–368, 1972.

[Gem74] Gembicki, F. W. : *Vector optimization for control with performance and parameter sensitivity indices*. Thèse de doctorat, Case Western Reserve University, Cleveland, OH, 1974.

[Geo68] Geoffrion, A. M. : Proper efficiency and the theory of vector maximization. *Journal of Mathematical Analysis and Applications*, 22(3):618–630, juin 1968.

[GG63] Gilmore, P. C. and Gomory, R. E. : A linear programming approach to the cutting stock problem–part ii. *OPERATIONS RESEARCH*, 11(6):863–888, 1963.

[GL97] Glover, F. and Laguna, M. : *Tabu Search*. Kluwer Academic Publishers, Norwell, MA, USA, 1997.

[Gol70] Goldfarb, D. : A family of variable metric updates derived by variational means. *Mathematics of Computation*, (24):23–26, 1970.

[Gol89a] Goldberg, D. E. : *Genetic algorithms in search, optimisation and machine learning*. Addison Wesley, Reading, 1989.

[Gol89b] Goldberg, D. E. : *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1989.

[HHF75] Haimes, Y., Hall, W., and Freedman, H. : *Multiobjective Optimization in Water Resources Systems : The Surrogate Worth Trade- offMethod*. Elsevier Scientific Publishing Company, 1975.

[Hol92] Holland, J. H. : *Adaptation in Natural and Artificial Systems : An Introductory Analysis with Applications to Biology, Control and Artificial Intelligence*. MIT Press, Cambridge, MA, USA, 1992.

[HWLL02] Hong, T.-P., Wang, H.-S., Lin, W.-Y., and Lee, W.-Y. : Evolution of appropriate crossover and mutation operators in a genetic process. *Applied Intelligence*, 16(1):7–1–7, 2002.

[HYB07] Huyao, L., Yuanjun, H., and Bennell, J. A. : The irregular nesting problem : a new approach for nofit polygon calculation. *Journal of the Operational Research Society*, 58(9):1235–1245(11), 2007.

[Jac10] Jacquenot, G. : *Méthode générique pour l'optimisation d'agencement géométrique et fonctionnel*. Thèse de doctorat, Ecole Centrale de Nantes, IRCCyN (France), 2010.

[Jak96] Jakobs, S. : On genetic algorithms for the packing of polygons. *European Journal of Operational Research*, 88(1):165–181, January 1996.

[JBMW09] Jacquenot, G., Bennis, F., Maisonneuve, J.-J., and Wenger, P. : 2d multi-objective placement algorithm for free-form components. *In Proceedings of ASME 2009 International Design Engineering Technical Conferences & Computers and Information in Engineering Conference IDETC/CIE 2009*, 2009.

[JP07] Jimeno, A. and Puerta, A. : State of the art of the virtual reality applied to design and manufacturing processes. *The International Journal of Advanced Manufacturing Technology*, 33:866–874, 2007.

[JR08] Jilkova, J. and Raida, Z. : Influence of multiple crossover and mutation to the convergence of genetic optimization. *In MIKON 2008, XVII International Conference on Microwaves, Radar and Wireless Communications in Poland*, 2008.

[KGV83] Kirkpatrick, S., Gelatt, C. D., and Vecchi, M. P. : Optimization by simulated annealing. *Science*, 220:671–680, 1983.

[KPP04] Kellerer, H., Pferschy, U., and Pisinger, D. : *Knapsack Problems*. Springer, Berlin, Germany, 2004.

[KPS08] Kelly, J., Papalambros, P. Y., and Seifert, C. M. : Interactive genetic algorithms for use as creativity enhancement tools. *Artificial Intelligence*, 2008.

[KRB10] Kefi, M., Richard, P., and Barichard, V. : Interactive configuration of restricted spaces using virtual reality and constraint programming techniques. *In Proceedings of the International Conference on Computer Graphics Theory and Applications, GRAPP*, pages 384–389, 2010.

[KTA04] Kamalian, R. R., Takagi, H., and Agogino, A. M. : Optimized design of mems by evolutionary multi-objective optimization with interactive evolutionary computation. *In GECCO (2)*, volume 3103 de *Lecture Notes in Computer Science*, pages 1030–1041. Springer, 2004.

[LB94] Landon, M. and Balling, R. : Optimal packing of complex parametric solids according to mass property criteria. *Journal of Mechanical Design*, 116:375–381, 1994.

[LCS06] Lewis, K., Chen, X., and Schmidt, L. : *Decision making in engineering design*. 2006.

[LG98] Lin, M. C. and Gottschalk, S. : Collision detection between geometric models : A survey. *In Proceedings of IMA Conference on Mathematics of Surfaces*, pages 37–6, 1998.

[Med96] Medjdoub, B. : *Méthode de conception fonctionnelle en architecture - Une approche CAO basée sur les contraintes : ARCHIPLAN*. Thèse de doctorat, Ecole Centrale Paris, LGI (Paris), 1996.

[Mie99] Miettinen, K. : *Nonlinear Multiobjective Optimization*, volume 12 de *International Series in Operations Research and Management Science*. Kluwer Academic Publishers, Dordrecht, 1999.

[MM95] Miettinen, K. and Mäkelä, M. M. : Interactive bundle-based method for nondifferentiable multiobjective optimization : NIMBUS. *Optimization*, 34:231–246, 1995.

[MM05] Mattson, C. A. and Messac, A. : Pareto frontier based concept selection under uncertainty, with visualization. *Optimization and Engineering*, 6(1):85–15, 2005.

[MP02] Michalek, J. J. and Papalambros, P. Y. : Interactive design optimization of architectural layouts. *Engineering Optimization*, 34:485–501, 2002.

[Nag95] Nagamachi, M. : Kansei engineering : A new ergonomic consumer-oriented technology for product development. *International Journal of Industrial Ergonomics*, 15:3–11, 1995.

[NF10] Nadeau, J.-P. and Fischer, X. : *Research in Interactive Design (Vol. 3) : Virtual, Interactive and Integrated Product Design and Manufacturing for Industrial Innovation*. 2010.

[NW99] Nocedal, J. and Wright, S. J. : *Numerical optimization*. Springer, août 1999.

[OTR03] Oduguwa, V., Tiwari, A., and Roy, R. : Handling integrated quantitative and qualitative search space in a real world optimisation problem. *Proceedings of the 2003 Congress on Evolutionary Computation CEC2003*, 2:1222–1229, 2003.

[Par96] Pareto, V. : *Cours d'économie politique : professé à l'Université de Lausanne*. Numéro vol. 1. F. Rouge, 1896.

[PCWB00] Parmee, I., Cvetkovic, D., Watson, A., and Bonham, C. : Multi-objective satisfaction within an interactive evolutionary design environment. *Journal of Evolutionary Computation, MIT*, 8 (2):197–222, 2000.

[Pis02] Pisinger, D. : Heuristics for the container loading problem. *European Journal of Operational Research*, 141(2):382–392, September 2002.

[Poi05] Poirson, E. : *Prise en compte des perceptions de l'utilisateur en conception de produit. Application aux instruments de musique de type cuivre.* Thèse de doctorat, Ecole Centrale de Nantes, IRCCyN (France), 2005.

[Pol03] Poles, S. : Moga-ii, an improved multi-objective genetic algorithm. *ESTECO, Technical Report 2003-006*, pages 1–16, 2003.

[PPB+11] Poirson, E., Petiot, J.-F., Bénabès, J., Boivin, L., and Blumenthal, D. : Detecting design trends using perceptive tests based on an interactive genetic algorithm. *In Proceedings of the IDETC-CIE conference, Washington, USA*, 2011.

[Roy85] Roy, B. : *Méthodologie multicritère d'aide à la décision*. Economica, Paris, 1985.

[Saa77] Saaty, T. L. : A scaling method for priorities in hierarchical structures. *Journal of Mathematical Psychology*, 15(3):234–281, juin 1977.

[Saa00] Saaty, T. L. : *Fundamentals of the Analytic Hierarchy Process*. RWS Publications, 4922 Ellsworth Avenue, Pittsburgh, PA 15413, 2000.

[Sak78] Sakawa, M. : Multiobjective optimization by the surrogate worth trade-off method. *Reliability, IEEE Transactions on*, R-27(5), décembre 1978.

[SC83] Steuer, R. and Choo, E.-U. : An interactive weighted tchebycheff procedure for multiple objective programming. *Mathematical Programming*, 26:326–344, 1983. 10.1007/BF02591870.

[SC95] Szykman, S. and Cagan, J. : A simulated annealing-based approach to three-dimensional component packing. *Journal of Mechanical Design*, 117:308–314, 1995.

[SC97] Szykman, S. and Cagan, J. : Constrained three-dimensional component layout using simulated annealing. *Journal of Mechanical Design*, 119:28–35, 1997.

[SC00] Su, Y. and Cagan, J. : An extended pattern search algorithm for three-dimensional component layout. *Journal of Mechanical Design*, 122:102–108, 2000.

[Sch95] Schott, J. R. : Fault tolerant design using single and multicriteria genetic algorithm optimization. Mémoire de D.E.A., Massachusetts Institute of Technology, Department of Aeronautics and Astronautics, Cambridge, Massachusetts, 1995.

[Sec98] Sechen, C. : *VLSI Placement and Global Routing Using Simulated Annealing*. Kluwer Academic Publishers, 1998.

[Sha70] Shanno, D. F. : Conditioning of quasi-newton methods for function minimization. *Mathematics of Computation*, (24):647–656, 1970.

[SKK99] Shibuya, M., Kita, H., and Kobayashi, S. : Integration of multi-objective and interactive genetic algorithms and its application to animation design. *In Proceedings of IEEE Systems, Man and Cybernetics (SMC)*, pages 646–651, 1999.

[SLGL00] Sanchez, S., Leroux, O., Gaildrat, V., and Luga, H. : Résolution d'un problème d'aménagement spatial à l'aide d'un algorithme génétique. *In AFIG'00, Grenoble*, 2000.

[SSO+74] Stone, H., Sidel, J., Oliver, S., Woolsey, A., and Singleton, R. C. : Sensory evaluation by quantitative descriptive analysis. *Food Technology*, 28(11):24–34, novembre 1974.

[ST95] Scheithauer, G. and Terno, J. : A branch & bound algorithm for solving one-dimensional cutting stock problems exactly. *Applicationes Mathematicae*, 23:151–167, 1995.

[SY99] Sait, S. M. and Youssef, H. : *Iterative Computer Algorithms with Applications in Engineering : Solving Combinatorial Optimization Problems*. IEEE Computer Society Press, Los Alamitos, CA, USA, 1st édition, 1999.

[SYS+04] Stump, G., Yukish, M., Simpson, T. W., , and O'Hara, J. : Trade space exploration of satellite datasets using a design by shopping paradigm. *In IEEE Aerospace Conference Proceedings*, 2004.

[SYSH03] Stump, G. M., Yukish, M., Simpson, T. W., and Harris, E. N. : Design space visualization and its application to a design by shopping paradigm. *In Proceedings of ASME 2003 International Design Engineering Technical Conferences & Computers and Information in Engineering Conference IDETC/CIE 2003*, volume 2003, pages 795–804. ASME, 2003.

[TFD11] Tiwari, S., Fadel, G., and Deb, K. : Amga2 : improving the performance of the archive-based micro-genetic algorithm for multi-objective optimization. *Journal of Engineering Optimization*, 43(4):377–401, 2011.

[TFF08] Tiwari, S., Fadel, G. M., and Fenyes, P. : A fast and efficient three dimensional compact packing algorithm for free-form objects. *In Proceedings of ASME International Design Engineering & Computers and Information in Engineering Conference (ASME DETC/CIE), New-York*, 2008.

[TSLL01] Teng, H.-f., Sun, S.-l., Liu, D.-q., and Li, Y.-z. : Layout optimization for the objects located within a rotating vessel - a three-dimensional packing problem with behavioral constraints. *Comput. Oper. Res.*, 28:521–535, May 2001.

[Ver92] Vern, J. : La logique floue : concepts et définitions. *Electronique Radio Plans*, 1992.

[VV99] Van Veldhuizen, D. A. : *Multiobjective evolutionary algorithms : classifications, analyses, and new innovations*. Thèse de doctorat, Wright Patterson AFB, OH, USA, 1999. AAI9928483.

[WHS07] Wäscher, G., Haubner, H., and Schumann, H. : An improved typology of cutting and packing problems. *European Journal of Operational Research*, 183:1109–1130, 2007.

[WT05] Wang, S. and Takagi, H. : Evaluation of user fatigue reduction through iec rating-scale mapping. *In* Abraham, A., Dote, Y., Furuhashi, T., Köppen, M., Ohuchi, A., and Ohsawa, Y., éditeurs : *Soft Computing as Transdisciplinary Science and Technology*, volume 29 de *Advances in Soft Computing*, pages 672–681. Springer Berlin - Heidelberg, 2005.

[WWC05] Wang, L.-h., Wei, P.-y., and Chang, Y.-t. : Reducing evaluation fatigue in interactive evolutionary algorithms by using an incremental learning approach. *In* Abraham, A., Dote, Y., Furuhashi, T., Köppen, M., Ohuchi, A., and Ohsawa, Y., éditeurs : *Soft Computing as Transdisciplinary Science and Technology*, volume 29 de *Advances in Soft Computing*, pages 629–640. Springer Berlin - Heidelberg, 2005.

[YFG08] Yi, M., Fadel, G. M., and Gantovnik, V. B. : Vehicle configuration design with a packing genetic algorithm. *International Journal of Heavy Vehicle Systems*, 15:433–448, 2008.

[Yin00] Yin, S. : *A computational framework for automated product layout synthesis based on extended pattern search algorithm.* Thèse de doctorat, Carnegie Mellon University, Pittsburgh, 2000.

[YP02] Yannou, B. and Petiot, J.-F. : Needs, perceptions, functions and products : highlight on promising design methods linking them. *In Proceedings of the IDMME Conference, Clermont-Ferrand, France*, 2002.

[YP04] Yoshimura, M. and Papalambros, P. Y. : Kansei engineering in concurrent product design : A progress review. *In Proceedings of the TMCE 2004, Lausanne, Switzerland*, 2004.

[ZCZ05] Zhang, J., Chung, H. S. H., and Zhong, J. : Adaptive crossover and mutation in genetic algorithms based on clustering technique. pages 1577–1578, 2005.

[ZDT00] Zitzler, E., Deb, K., and Thiele, L. : Comparison of multiobjective evolutionary algorithms : Empirical results. *Evolutionary Computation*, 8:173–195, 2000.

[ZhYf03] Zhi-hua, L. and Yi-fang : Virtual facility layout design using virtual reality techniques. *Whuan University Journal of Natural Sciences*, 8:41–45, 2003.

[Zit99] Zitzler, E. : *Evolutionary Algorithms for Multiobjective Optimization : Methods and Applications.* Thèse de doctorat, ETH Zurich, Switzerland, 1999.

[ZLT02] Zitzler, E., Laumanns, M., and Thiele, L. : Spea2 : Improving the strength pareto evolutionary algorithm for multiobjective optimization. *In IEEE Congress on Evolutionary Computation*, 2002.

[ZTS08] Zhang, B., Teng, H.-F., and Shi, Y.-J. : Layout optimization of satellite module using soft computing techniques. 8(1):507–521, 2008.

Liste des figures

Liste des tableaux

Optimisation intégrée et interactive pour l'agencement d'espace

Résumé

Influençant fortement la conception de nombreux produits et systèmes industriels, l'optimisation d'agencement est au cœur des problématiques de recherche scientifique. L'élaboration d'une solution d'agencement optimale est rendue d'autant plus complexe qu'il s'agit de répondre aux exigences croissantes des concepteurs travaillant sur des projets divers.
Ce travail de doctorat propose une méthode intégrée visant à résoudre un problème d'optimisation d'agencement d'espace, depuis le besoin formulé par le concepteur jusqu'à l'obtention d'une solution idéale. Cette méthode, générique et interactive, s'appuie sur un processus en quatre étapes : la description, la formulation, la résolution du problème et la prise de décision finale. Les deux premières étapes consistent à écrire le problème d'optimisation d'agencement en prenant en compte les différentes exigences du concepteur. Ensuite, l'étape de résolution s'appuie sur l'utilisation d'une stratégie d'optimisation multiobjectif et modulaire. Enfin, la dernière étape consiste, pour le concepteur, à faire un choix de conception sur les solutions proposées par l'algorithme.
Afin de matérialiser et valider la méthode, deux problèmes industriels d'agencement d'espace sont résolus. Le premier cas d'étude, proposé par l'entreprise Thales Communications & Security, porte sur le placement optimal d'équipements dans un shelter (abri technique mobile). Le deuxième problème, plus sommaire, étudie le stockage de composants dans un container. Enfin, le travail réalisé au cours de ce doctorat a permis le développement d'un logiciel qui reprend les différentes étapes de la démarche d'optimisation.

Mots-clés : agencement d'espace, optimisation multiobjectif, interactivité, aide à la décision.

Abstract

Having a significant impact on the design of many products and industrial systems, layout design optimization is at the heart of scientific issues. The design of an optimal layout solution is a critical and complex task due to the increasing demands of designers working on varied projects.
This PhD work proposes an integrated approach to solving layout optimization problems, from the needs expressed by the designer to the creation of an ideal solution. This generic and interactive method is based on a design process divided into four steps : the description, formulation, and solving of the problem, and the final decision. The first two steps consist in writing the optimization problem, taking into account the different requirements of the designer. Then, the problem solving step is based on a multiobjective modular optimization strategy that combines a genetic algorithm with local optimization modules. The final step allows the designer to make a final decision on the optimal solutions proposed by the algorithm.
In order to materialize and validate the method, two industrial layout problems are solved. The first one is proposed by Thales Company and deals with the search for an optimal layout of facilities in a shelter. Treated in two and three dimensions, the problem is a global test for the method. The second application, which is more basic, studies the storage of components in a container.
The work performed during this PhD has enabled the development of a tool for the designer. Handling components with a parallelepiped shape, the software includes the different steps of the optimization method and integrates the multiobjective solving of layout optimization problems.

Keywords : layout problems, multiobjective optimization, interactive decision making.

Discipline : Sciences de l'Ingénieur